# Computational Techniques to Identify Rare Events in Spatio-temporal Data

**A THESIS**
**SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL**
**OF THE UNIVERSITY OF MINNESOTA**
**BY**

**Varun Mithal**

**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS**
**FOR THE DEGREE OF**
**DOCTOR OF PHILOSOPHY**

**Prof. Vipin Kumar, Advisor**

**May, 2018**

# Acknowledgements

I want to thank many people who made my time in graduate school fruitful and memorable. I am extremely grateful to my advisor Prof. Vipin Kumar for his support and guidance throughout my PhD. I consider myself fortunate to have studied data science under his supervision. His passion towards research and his enthusiasm to solve difficult problems are infectious and have been the key sources of motivation during my thesis work.

I am grateful to my thesis committee members- Prof. Arindam Banerjee, Prof. Daniel Boley, and Prof. Snigdhanshu Chatterjee for their valuable suggestions and feedback that have helped in shaping my dissertation.

I immensely cherish the time spent with my fellow students in graduate school including my seniors Varun Chandola, and Shyam Boriah, and my peers Ankush Khandelwal, Anuj Karpatne, Ashish Garg, Guruprasad Nayak, Ivan Brugere, James Faghmous, Jaya Kawale, Saurabh Agarwal, Vikrant Krishna, and Xi Chen. This thesis work would not have been possible without their collaboration.

I also thank my friends- Anurag Kumar, Nipun Garg, Surabhi Mithal, Zainab Hazarika, and Zoheb Borbora- who made my graduate school journey a fun ride.

Finally, I thank my parents and family for patiently supporting me in my academic pursuits.

# Dedication

This thesis is dedicated to my grandparents.

## Abstract

Recent attention on the potential impacts of land cover changes to the environment as well as long-term climate change has increased the focus on automated tools for global-scale land surface monitoring. Advancements in remote sensing and data collection technologies have produced large earth science data sets that can now be used to build such tools. However, new data mining methods are needed to address the unique characteristics of earth science data and problems. In this dissertation, we explore two of these interesting problems, which are (1) build predictive models to identify rare classes when high quality annotated training samples are not available, and (2) classification enhancement of existing imperfect classification maps using physics-guided constraints.

We study the problem of identifying land cover changes such as forest fires as a supervised binary classification task with the following characteristics: (i) instead of true labels only imperfect labels are available for training samples. These imperfect labels can be quite poor approximation of the true labels and thus may have little utility in practice. (ii) the imperfect labels are available for all instances (not just the training samples). (iii) the target class is a very small fraction of the total number of samples (traditionally referred to as the rare class problem). In our approach, we focus on leveraging imperfect labels and show how they, in conjunction with attributes associated with instances, open up exciting opportunities for performing rare class prediction. We applied this approach to identify burned areas using data from earth observing satellites, and have produced a database, which is more reliable and comprehensive (three times more burned area in tropical forests) compared to the state-of-art NASA product.

We explore approaches to reduce errors in remote sensing based classification products, which are common due to poor data quality (eg., instrument failure, atmospheric interference) as well as limitations of the classification models. We present classification enhancement approaches, which aim to improve the input (imperfect) classification by using some implicit physics-based constraints related to the phenomena under consideration. Specifically, our approach can be applied in domains where (i) physical properties can be used to correct the imperfections in the initial classification products, and (ii) if clean labels are available, they can be used to construct the physical properties.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Advancements in remote sensing and data collection technologies have produced large-scale earth science data sets that can be used to build automated tools for global-scale land surface monitoring. One of the unique oportunity that these massive and information rich datasets offer is to advance our understanding of land cover change, climate change and their anthropogenic impacts. In particular, remote sensing data offers the potential for reliable and timely land surface monitoring, and identify land cover changes such as disturbances (eg. forest fires, deforestation) and conversions (eg. urban growth, changes in water levels) [1, 2] at a global scale. However, earth science data sets pose unique challenges that require advances in the field of spatio-temporal data mining for these automated tools to be effective. In particular, data mining approaches need to address challenges posed by rarity of the change events, lack of labeled samples for supervision, noise and missing data due to clouds and other atmospheric interference, seasonal and spatial variations in the data. This thesis is a step in the direction of addressing these challenges in order to advance the state-of-art in computational approaches for land surface monitoring. This chapter provides a brief overview of the computational methods being explored, the thesis statement and organization together with a summary of contributions.

## 1.1 Computational Methods for Land Surface Monitoring

We present a brief overview of the two approaches for land surface change monitoring developed in this thesis, and also discuss some of the key challenges associated with each of them.

### 1.1.1 Predictive model to identify change events

These methods build a classification model on attributes such as spectral features to predict the probability of occurrence of a change event. There are three key challenges in applying these methods to earth science. First, the target class (change events) are a very small fraction of the total number of samples. Second, supervised methods require labeled samples of the change event for training the classification models. But expert annotated high quality labeled samples are difficult to obtain, especially due to the rarity of the change events in the data. Third, a single model does not perform well globally due to the variations in soil, climate, and land cover. One approach to address this spatial heterogeneity is to partition data into smaller homogeneous units and train separate models for each homogeneous unit. However, such partitioning further exacerbates the issue of paucity of labeled samples.

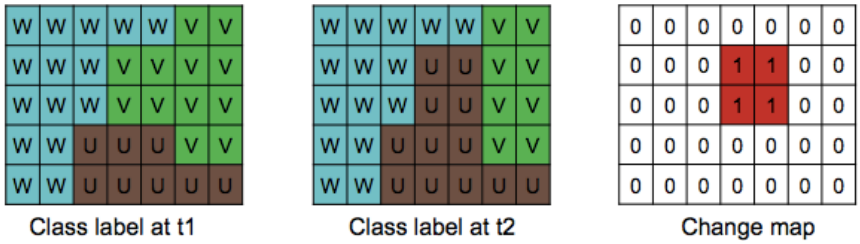



Figure 1.1: Post-classification comparison uses the classified maps from two time steps, and assigns pixels to change or not change category based on changes in their class labels at the two time steps. In this example, we show a caricature of images at two time steps with each pixel assigned to either Water (W), Urban (U) or Vegetation (V) land class. The change map is derived using post-classification comparison of the two images.

### 1.1.2 Post-classification comparison

These methods leverage existing classified land cover maps for change detection. First, machine learning models coupled with manual tuning are used to obtain land cover maps, which classify each pixel to a land cover category, from the raw multi-spectral data. Change detection is then performed by comparing two snapshots of classification maps separated in time (see Figure 1.1). The accuracy of change detection for these methods is impacted by the errors in the base classification maps used. To illustrate how classification accuracy impacts change detection, let us consider a classifier built for a data set with two target classes and balanced training samples. Assume the classifier assigns a given test object to the incorrect class with probability $\epsilon$ and to the correct class with probability $1-\epsilon$. Let us also assume the fraction of land surface that actually changed in this period is $p$. Due to classification error, $\epsilon$ fraction of pixels belonging to class *1* will be assigned class *2* in time $t_2$. Similarly, $\epsilon$ fraction of pixels belonging to class *2* will be assigned class *1* in time $t_1$. Thus, even when there is *no* land cover change between $t_1$ and $t_2$, $\epsilon$ fraction of class *1* pixels and $\epsilon$ fraction of class *2* pixels will be designated as changes from class *1* to class *2* and vice-versa. These incorrect labels will contribute to the false positives for a change detection query. Similarly, there will be $2\epsilon p$ false negatives. Ignoring the higher order terms in $\epsilon$, the expected recall is $\frac{p-2\epsilon p}{p}$, and the expected precision is $\frac{p-2\epsilon p}{p+2\epsilon}$. However, changes in land cover typically occur in a very small portion of a large region of study; the area changed is often less than 1% of the total area ($p \approx 0.01$). Therefore, recall $\approx 0.8$, and precision $\approx 0.05$. Thus, even for high accuracy, state-of-the-art land cover classification products, the precision of change detection maps can be as poor as 5%. The analysis above shows that when land cover change mapping is done using post-classification comparison of images, even small amounts of classification inaccuracy can significantly lower precision. However, ensuring high accuracy in land cover classification is challenging because remote sensing data is often plagued with noise and missing data due to atmospheric interference.

## 1.2 Thesis Statement

In this thesis, we address some of the challenges associated with applying computational techniques to detect change events in spatio-temporal data. We apply these techniques

to identify land changes including forest fires, urban growth, and lake water level changes from remote sensing data.

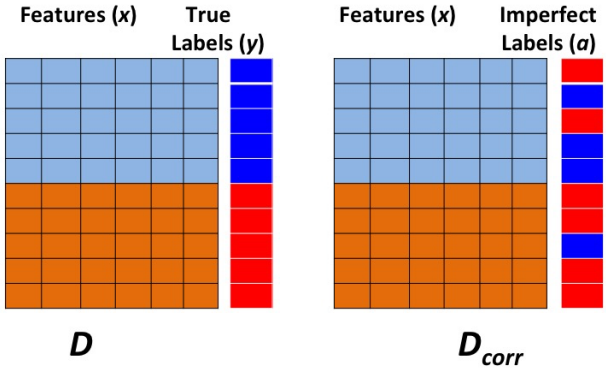



Figure 1.2: Training dataset $D$ (with true labels $y$) and training dataset $D_{corr}$ (with imperfect labels $a$).

- *Learn predictive models to identify rare events in the complete absence of expert annotated training data.* Our focus is on building predictive models in problem settings with the following characteristics: (1) instead of true labels only imperfect labels are available for training samples (an example of true and imperfect labels is shown in figure 1.2). These imperfect labels can be quite poor approximation of the true labels and thus may have little utility in practice. (2) the imperfect labels are available for all instances (not just the training samples). (3) the target class is a very small fraction of the total number of samples (traditionally referred to as the rare class problem).

- *Enhance imperfect classification maps of multi-temporal gridded data plagued by noise and missing data that may be auto-correlated in space and time.* Our goal is to improve the input (imperfect) classification by using some implicit physics-based constraints related to the phenomena under consideration. Specifically, our approaches can be applied in domains where (i) physical properties can be used to correct the imperfections in the initial classification products, and (ii) if clean

labels are available, they can be used to construct the physical properties. Figure 1.3 shows a schematic of classification enhancement.

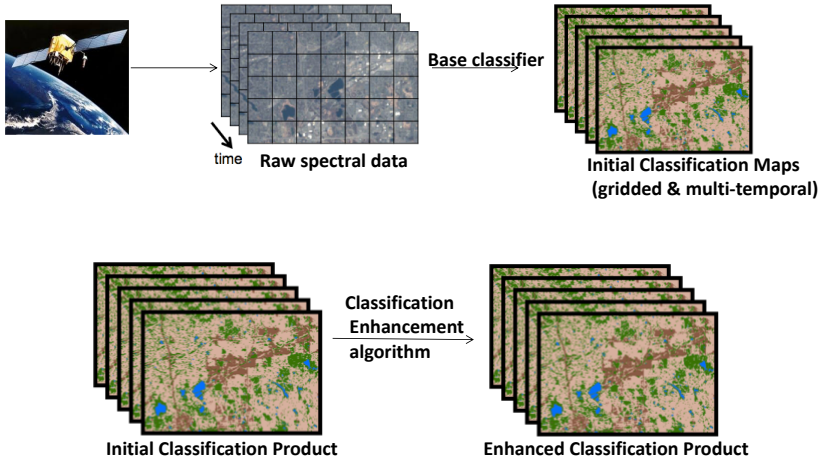



Figure 1.3: A schematic of the Classification Enhancement process. First, a base classification model is used to classify pixels of the raw images from the satellite into land cover classes. Then a classification enhancement approach is deployed on the intial classification maps to improve the classification accuracy using physical properties.

## 1.3 Thesis Contributions and Organization

This thesis presents algorithms to identify rare events in spatio-temporal data sets. Also, we have shown the efficacy of these algorithms to solve important societal problems in earth science domain. The main contributions made in this thesis are as follows:

- Chapter 2 presents a novel approach to train predictive models to identify rare classes under situations when expert-annotated labels are not available for training samples [3]. The approach makes use of imperfectly labeled samples in a principled manner to train classification models. We show that under certain assumptions on the label noise the algorithm presented can accurately maximize precision and

recall for the rare class. We evaluate the proposed approach on two real world problems- identifying forest fires [4] and urban growth from remote sensing multi-spectral data.

- Chapter 3 presents a classification enhancement approach that uses temporal context to reduce noise and missing data in urban maps by making use of the fact that urban growth is a rare and slow phenomena. Moreover, we show that a longer temporal context can also be used to discover new land classes such as mixed vegetation types, which often contribute to a considerable portion of change detection errors in bi-temporal classification comparison maps [5].

- Chapter 4 further builds on the idea of classification enhancement to improve quality of remote sensing classification maps by leveraging the constraints imposed by the physical properties of the phenomenon being studied. We present an approach that uses elevation based constraints to improve classification maps for lake water bodies. Furthermore, this approach demonstrates that data can also be leveraged to reconstruct parameters of physical constraints, eg. in the example of lake water bodies we can reconstruct the relative depth order of pixels from multi-temporal land cover classification maps.

- We summarize the contributions of this thesis and discuss some future research directions in Chapter 5.

# Chapter 2

# RAPT: Rare Class Prediction in Absence of True Labels

## 2.1 Motivation

A supervised learning task involves building a classification model that maps the input feature vectors $\mathbf{x}$ to a target class $y$ using a training data $D = \{\mathbf{x}, y\}_n$ of representative samples and their corresponding labels.

In contrast, our focus is on building classification models in problem settings with the following characteristics: (1) instead of true labels only imperfect labels are available for training samples. These imperfect labels can be quite poor approximation of the true labels and thus may have little utility in practice. (2) the imperfect labels are available for all instances (not just the training samples). (3) the target class is a very small fraction of the total number of samples (traditionally referred to as the rare class problem).

Characteristics 1 and 3 (even if taken individually) can challenge most of the traditional supervised learning methods. Recently, approaches have been proposed to learn classification models with imperfect labels [6, 7]. However, these approaches have been designed for balanced class settings, and cannot be directly applied in rare class scenarios (see Figure 2.1). Most existing rare class algorithms address the problem of very few positive samples by oversampling (or SMOTE [8] style synthetic instance generation) the positive class. For our scenario, this is not the case, as we have plenty of positive

samples (despite the skew) because all records have (imperfect) labels, and the key issue to be addressed is to train classification models rare class scenarios using only imperfect labels. We present a three step framework- RAre class Prediction in absence of True labels (RAPT)- for problem settings with the above three characteristics. The first step of the RAPT framework learns a classifier by only using imperfectly labeled training samples. We show that, under certain assumptions on the imperfect labels, the quality of this classifier can be almost as good as the one constructed using true labels. The second and third steps of the RAPT framework make use of the fact that imperfect labels are available for all instances to address the challenges due to rarity of the target class. Before providing further details on this framework, we present two practical problems of global significance that require a predictive model under the scenario described above as well as existing work in machine learning that has addressed some of these challenges and has also served as an inspiration for the solution presented.

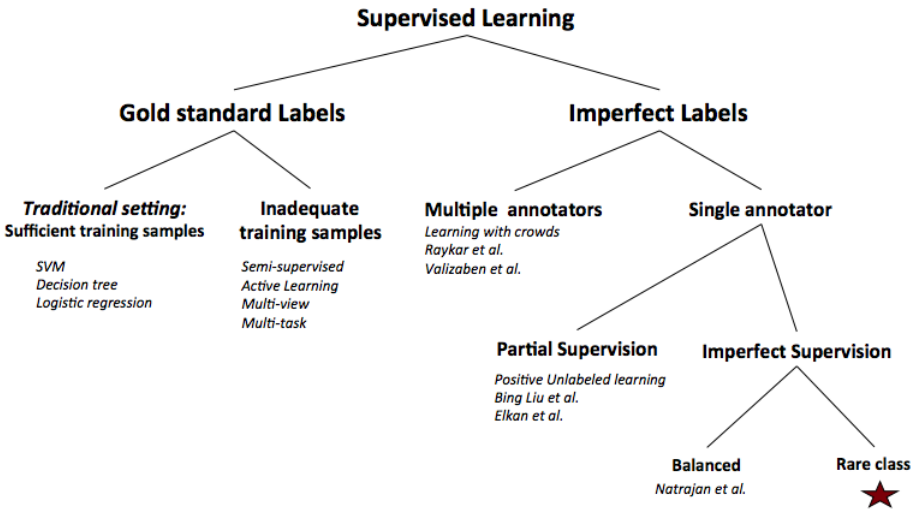



Figure 2.1: Taxonomy of research related to learning in absence of labeled data.

### 2.1.1 Motivating problems

Forest fires are a major source of greenhouse gas emissions and have a significant footprint on the flora and fauna, and the air quality of the region [9]. Thus, there is a need for accurate and cost-effective fire mapping techniques that provide earth scientists with

the spatial extent and timing of fire events for understanding biomass burning and its impact on the global climate system. Spectral data from earth observing satellites can be used to address this problem if a predictive model can be built to map spectral observations of a location to the target label (burned/not burned) [10]. The machine learning task of building predictive models on spectral data to identify fire events exhibits the three characteristics that are the focus of this work. First, fire monitoring is a rare class problem and the number of fires in forested areas is usually a tiny fraction of the total number of locations. As an example, California state has about 70,000 sq. km. of forests and in year 2008, which is one of the worst years for fires in California in the last decade, 2,200 sq. km. of this area was burned (i.e., only about 3% of total locations). Second, high quality annotated training samples for fires are available in only some parts of the world (eg. several states in U.S.A. and Canada). A model trained on samples from these regions can lead to poor performance in other regions of the world, as the relationship between the explanatory variables and target variable greatly varies in space and time [11, 12]. Moreover, obtaining high quality annotated training samples in all combinations of regions and seasons is prohibitively expensive at a global scale. However, active fire signal (a signature of thermal anomaly visible from earth observing satellites) is available for all locations and seasons for the entire period for which satellite data is available (around 16 years for MODIS [13]). Since biomass burning often causes an anomaly in thermal signal, active fire signal can be considered as an imperfect label while building a model for fires. It is important to note that this imperfect label by itself is of quite poor quality (e.g., it's recall and precision can be very low in many parts of the world [11]).

Another environmental problem that we study is mapping urban areas from satellite data, which has recently received attention because increased urbanization has impacted a host of environmental factors [14]. For this application, night-time light signal that measures the average intensity of illumination on land surface during the night is available for all locations from the earth observing satellites. Most urban locations are likely to show a higher night-time light intensity compared to non-urban locations and therefore night time lights can be used as an imperfect labels for urban area mapping. The machine learning task is to learn a classification model that maps the spectral observations to target class (urban/not urban) using the imperfect labels obtained from

night-time light signal.

### 2.1.2 Machine learning challenges and related work

Recently, there has been a lot of work in machine learning aimed at handling imperfectly-labeled training samples (i.e., characteristic 1). In this problem setting, only a noisy training data $D_{corr} = \{\mathbf{x}, a\}_n$ is available whose labels $a$ are corrupted versions of the actual ground truth $y$. Note that the noise in training data here refers to flips in target labels $y$, and not to the presence of noise or anomalies (outliers) in features vectors. One of the most widely studied label corruption is class conditional label noise (CCN) [6, 7, 15]. CCN label noise implies that the labels of the training samples have been flipped such that the flip probability depends only on the true class label and not on the attributes of the samples. The positive instances have been flipped with probability $\alpha$ and the negative instances have been flipped with probability $\beta$.

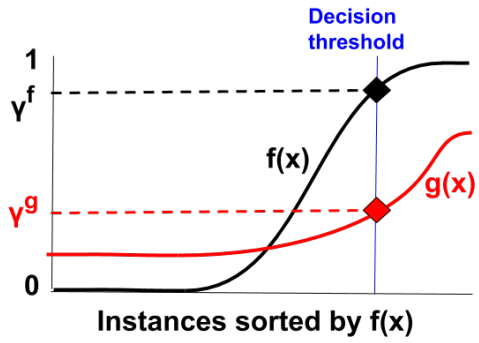



Figure 2.2: Figure shows the relationship between $f(x)$ and $g(x)$ for instances arranged on x-axis sorted by $f(x)$. $\gamma^g$ is the threshold on $g$ corresponding to $\gamma^f$ on $f$.

Under the assumption that $\alpha + \beta < 1$, Menon et al. [6] have shown that the true class probability $P(y = 1|\mathbf{x})$ and corrupted class probability $P(a = 1|\mathbf{x})$ have a monotonically increasing linear relationship. Now consider two functions $f(\mathbf{x})$ and $g(\mathbf{x})$, that have been selected from a hypothesis space using a machine learning algorithm, to model $P(y = 1|\mathbf{x})$ and $P(a = 1|\mathbf{x})$ (or some monotonic function of these probabilities) respectively. It is easy to see that these two functions will also be monotonically related. This monotonic relationship implies that the ranking of data instances sorted by $f(\mathbf{x})$


(a) thresholds do not coincide

(b) thresholds coincide

Figure 2.3: Caricature of performance measures computed using true labels $y$ (in black) and imperfect labels $a$ (in red).

and $g(\mathbf{x})$, trained on $D$ and $D_{corr}$ respectively, will be identical. Figure 2.2 shows a caricature of the relationship between $f(x)$ and $g(x)$ for a dataset sorted by $f(x)$.

In the traditional setting a decision threshold $\gamma^f$ is chosen on $f$ to perform classification; an instance $x_i$ is assigned to the positive class if $f(x_i)$ is greater than $\gamma^f$, otherwise it is assigned to the negative class. It can also be seen from Figure 2.2 that for any threshold $\gamma^f$ on $f(\mathbf{x})$, there exists a corresponding threshold $\gamma^g$ on $g(\mathbf{x})$ such that the two classifiers $(f, \gamma^f)$ and $(g, \gamma^g)$ give identical predictions. Proper choice of the threshold value depends on the performance measure $M$ that one is interested in optimizing (eg. classification accuracy, balanced error rate, F-measure and G-measure). A commonly used approach is to do a grid search over the candidate threshold values for optimizing $M$ on a validation dataset. Specifically, the value of $M$ is computed corresponding to each candidate threshold value, and the threshold value that optimizes $M$ is selected. Selecting the threshold $\gamma^g$ using this approach requires the ability to compute the value of $M$ on a validation dataset for each candidate threshold.

Since a validation dataset with true labels $y$ is not available in our setting, one possibility is to use the imperfect labels $a$ as target to compute $M$. However, this approach can lead to selection of an incorrect threshold, as the value of $M$ computed using $a$ can be quite different from the one computed using $y$. Figure 2.3(a) shows a caricature of a hypothetical measure $P$ computed using $y$ for different threshold values, and the corresponding value of $P$ computed using $a$. While selecting the threshold on $g$

that optimizes $P$ computed with $y$ as target will give a classifier that has performance similar to the classifier $(f, \gamma^f)$, selecting this threshold to optimize $P$ computed with $a$ as target can result in a classifier with considerably worse performance.

Despite this, as shown by Menon et al. [6] in some cases it is possible to optimize a performance measure $M$ just using imperfect labels. For example, if the values of $M$ computed using $y$ and $a$ are affinely related, then the *M-optimal* classifiers built on $D$ and $D_{corr}$ coincide. Such affine relationship has been shown to hold for some performance measures such as classification accuracy for balanced class problems [6, 7], balanced error rate [6], and area under the curve [6]. Figure 2.3(b) shows a caricature of classification accuracy computed using $a$ and $y$ for different threshold values. It can be seen from this figure that optimizing accuracy computed with $a$ as target will select the same threshold as optimizing accuracy computed with $y$ as target. *As a result, for such measures one can effectively treat CCN corrupted samples as if they are clean while training classifiers to optimize one of these performance measures.*

However, the focus is to learn classification models for rare class scenarios when imperfect labels are available (shown by *red star* in the Figure 2.1). If the target class is rare, performance measures such as classification accuracy and balanced error rate are not very effective. Instead, it is desirable to choose a decision threshold that optimizes a combined metric of the precision and recall of the rare class [16–19] such as harmonic mean (F-measure) or geometric mean (G-measure). However, these measures computed on $a$ are not affinely related to their counterpart computed on $y$. Hence, the performance of the classifier built on $D_{corr}$ to optimize G-measure (or F-measure) can be considerably worse than the classifier built on $D$.

To illustrate this, we trained a classifier on $D$ to optimize G-measure with true labels $y$ as target (*red curve* in Figure 2.4). Then, we trained classifiers to optimize G-measure with the imperfect labels $a$ as target using multiple $D_{corr}$, each with different degree of label noise (*blue curve* in Figure 2.4). The performance (measured as G-measure) is reported with respect to the true labels $y$. We observe that the performance of classifier trained on $D_{corr}$ is comparable to the classifier trained on $D$ for very low levels of label noise. However, as the level of label noise is increased, the performance of classifier trained on $D_{corr}$ degrades considerably.

Note that the special case of $\beta = 0$ (but non-zero $\alpha$) corresponds to the PU learning

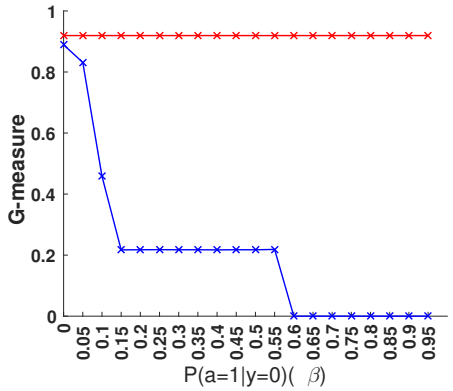



Figure 2.4: Performance of two classifiers on different training sets corresponding to different quality of imperfect labels (red: classifier built using true labels, blue: classifier built using corrupted labels but treating them as true label). The x-axis shows the imperfection (measured as $\beta$) in the training samples for each dataset. (The flip probability $\alpha$ is fixed at 0.4). The y-axis shows the performance (measured as G-measure) of each of the two classifiers for every dataset.

setting explored in previous studies [20–22]. These studies have shown that G-measure can be optimized for PU learning, as the G-measure computed on PU samples is affinely related to the G-measure computed on $y$. Note that this result does not hold for F-measure. The first step of RAPT uses a similar idea to train a classifier that optimizes G-measure in our problem setting where both $\alpha$ and $\beta$ are non-zero. In particular, we optimize a function that is affinely related to the product of precision and recall, and which can be estimated using only imperfect labels $a$.

Any classification model (even if trained on gold standard training samples) will have a non-zero false positive rate (FPR) and false negative rate (FNR) because of some overlap between the classes in the feature space. Thus, in practice even classifiers trained to optimize precision and recall using a sufficient number of hand-picked high quality training samples may suffer from poor precision and recall, especially when the imbalance between the classes is large [23]. We illustrate this issue by learning classifiers on 3 different datasets, that have identical data distribution in the feature space but differ in the skew between the positive and negative classes. Each classifier is trained to maximize the G-measure for the corresponding dataset using gold standard

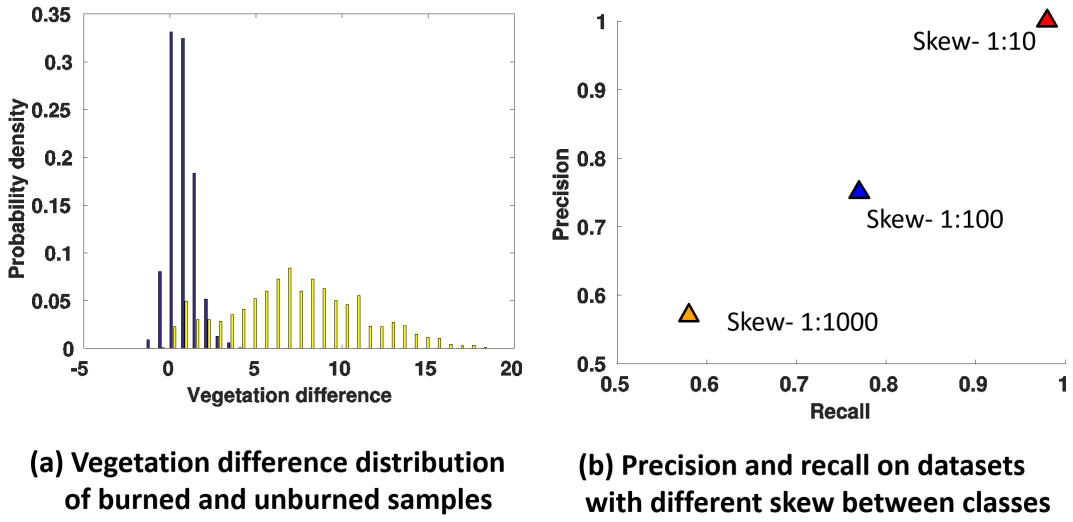



Figure 2.5: (a) An illustrative example of overlap between positive (burned) and negative (unburned) class in the feature space for the forest fire application. (b) The performance of classifiers trained on expert-annotated samples for this example corresponding to 3 datasets with different skew between the two classes. Figure is best viewed in color.

training samples. Figure 2.5 shows the performance of these classifiers on their respective datasets. We notice that the precision and recall obtained decrease as the imbalance of the dataset is increased. One approach to improve the precision and recall in such imbalanced class scenarios is to combine predictions obtained from multiple independent classifiers [23]. While there is much work on combining independent predictions to maximize accuracy in context of balanced classes [24–26], to the best of our knowledge there has been no work on combination methods to jointly maximize precision and recall, which is necessary for rare class scenarios and is the focus of RAPT stage 2 and 3.

### 2.1.3 Our Approach and Contributions

In the following, we present an overview and the key contributions of the three steps of the RAPT framework.

**Step 1:** In the first step of RAPT, we present a method to train a classifier using

imperfectly labeled samples $D_{corr}$ that, under the CCN conditions and some additional assumptions, is almost as effective for optimizing G-measure ($\sqrt{precision * recall}$) as the classifier trained using expert-annotated samples $D$. As we discussed earlier, G-measure computed on $a$ and $y$ are not affinely related, and hence the classifier built using $D_{corr}$ to optimize G-measure does not coincide with the classifier built on $D$ to optimize G-measure. Menon et al. [6] have shown that if $\alpha$ and $\beta$ are known, then estimates of many performance measures (including G-measure) computed on $y$ can be derived using only $a$. If $\alpha$ and $\beta$ can be estimated, then this result allows optimization of G-measure (according to true labels $y$) using only the imperfectly labeled samples from $D_{corr}$. We show that for rare class scenarios, $\beta$ can be robustly estimated under the mild assumption that there are a sufficient number of negative samples with $P(y = 1|x)$ as 0; however, the estimates of $\alpha$ are not robust. A key result proved in this step is that if $\beta$ is known, then a function that is affinely related to G-measure (according to the true labels $y$) can be estimated using only imperfectly labeled samples, which allows selection of a decision threshold on $g(\mathbf{x})$ that maximizes the G-measure. Thus, the method presented in step 1 advances the state-of-art in learning with noisy labels to rare class scenarios by optimizing G-measure instead of classification accuracy. Moreover, this method can be seen as a generalization over the algorithms to identify rare classes in PU learning setting (which make an additional assumption that flip probability of negatives is 0).

**Step 2:** In the second step of RAPT, we combine predictions from the function $g$ trained in step 1 and the imperfect labels to address the issue of poor performance due to high skew between the rare and majority class. In particular, we use a simple combination strategy that takes a *logical AND* of the two prediction sources, i.e., we define the step 2 classifier on the function $q(x) = a \times g(x)$. Figure 2.6 shows an illustrative example of the function $q(x)$ corresponding to a pair of $g(x)$ and $a$. Similar to the step 1, classification in step 2 is done using a decision threshold $\gamma$ on $q(x)$. For a given decision threshold $\gamma$, the classifier $(q, \gamma)$ eliminates many false positives of the classifier $(g, \gamma)$, while also eliminating some of its true positives. We show that for rare class scenarios, the gain in precision (due to elimination of false positive) is significantly higher compared to the loss in recall (due to elimination of true positives); thus resulting in an overall improvement in performance (G-measure). In particular, we prove that under the CCN assumption for any threshold $\gamma$, the ratio of G-measure of step 2 classifier

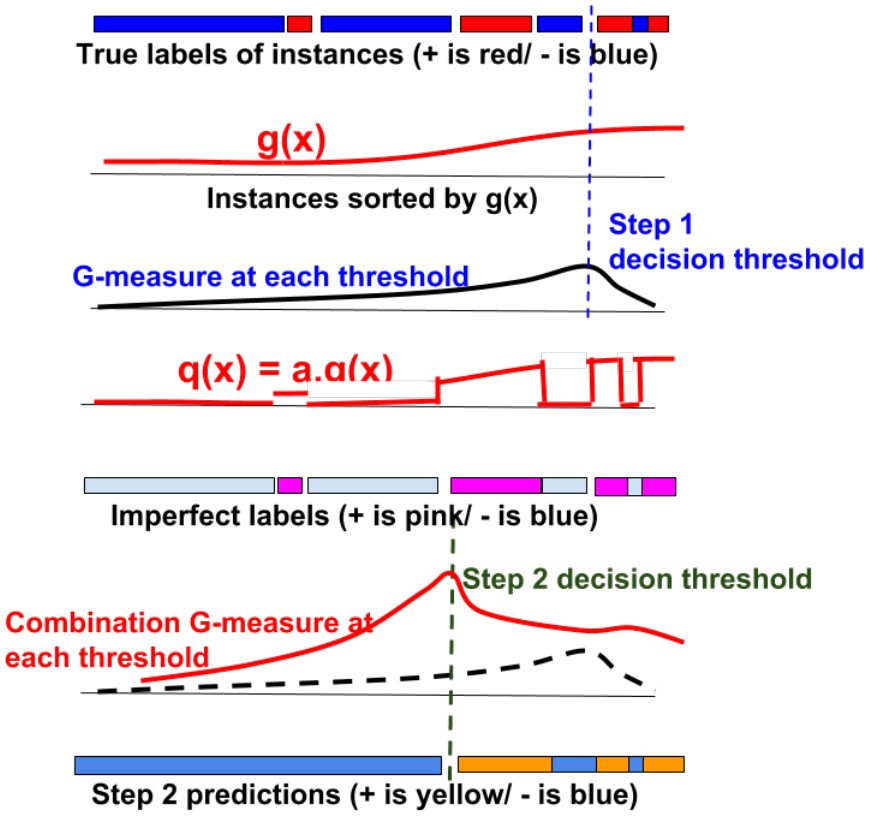



Figure 2.6: A caricature of the second step of RAPT.

$(q, \gamma)$ with respect to step 1 classifier $(g, \gamma)$ is $\frac{1-\alpha}{\sqrt{Pre^a_{step1}}}$, where $Pre^a_{step1}$ is the estimated precision of step 1 classifier computed on imperfect labels $a$. Note that a higher gain in G-measure is obtained after step 2 if the estimated precision of the step 1 classifier is low, which is precisely the situation under which we need a follow up step to improve the precision. It can also be seen from Figure 2.6 that the threshold that optimizes the G-measure of step 2 classifier on $q$ differs from the optimal threshold selected for the step 1 classifier on $g$. This is because of the elimination of many more false positives compared to true positive due to combination with imperfect labels, which allows lowering of the optimal threshold that jointly maximizes the precision and recall. In Step 2 of RAPT, we present a method to select the threshold on $q$ that optimizes the G-measure after the combination step using only the imperfect labels $a$. Note that using imperfect labels as a prediction source is possible only if they are available for all instances and

not just the training samples ($D_{corr}$). Hence, characteristic 2 is critical for applying stage 2 of RAPT.

**Step 3:** Increase in the precision of rare class observed in Step 2 of RAPT is achieved at the expense of some loss in recall. The focus of Step 3 in RAPT is to use collective classification methods [27] to improve the recall of the rare class by leveraging the guilt-by-association principle. Collective classification methods make use of the information present in the labels of the neighbors in addition to the observations associated with the individual instances while assigning the final label for each instance. This step of RAPT can be viewed as a collective classification method that uses a spatially smooth density function formed by step 2 output as the prior probability, which is then used in conjunction with the output of the classifier trained in step 1 to make the final prediction. This step brings in instances assigned to positive class by classifier of Step 1 (but not included in step 2 output), if they have a presence of confident positive instances identified in step 2 in their neighborhood. A flowchart showing the data flow in each of the three steps of RAPT is presented in Figure 2.7.

**Experimental evaluation** We evaluated the RAPT framework on two applications of mapping forest fire and urban extent from satellite data. Our results show that RAPT framework effectively learns models to identify the rare class instances by using imperfect labels, which by themselves have a precision and/or recall as poor as 0.6 for some of the datasets in these applications.

**Estimating model performance** In real-world problems it is also desirable to know the expected performance of the model on unseen test set. In general this is estimated using a hold-out data (or using $k$-fold cross validation) from the available training samples $D$. But in our case the precision and recall computed on $a$ do not match the desired precision and recall computed on $y$. We show that it is possible to compute a lower bound of the precision using imperfect labels. Note that optimizing G-measure ensures that both precision and recall are comparable at the decision threshold, and that one of the measures is not higher at the expense of the other. Hence, this lower bound on precision can be used to provide some indication of whether the model built by RAPT is expected to have a performance (both precision and recall) in the acceptable range.

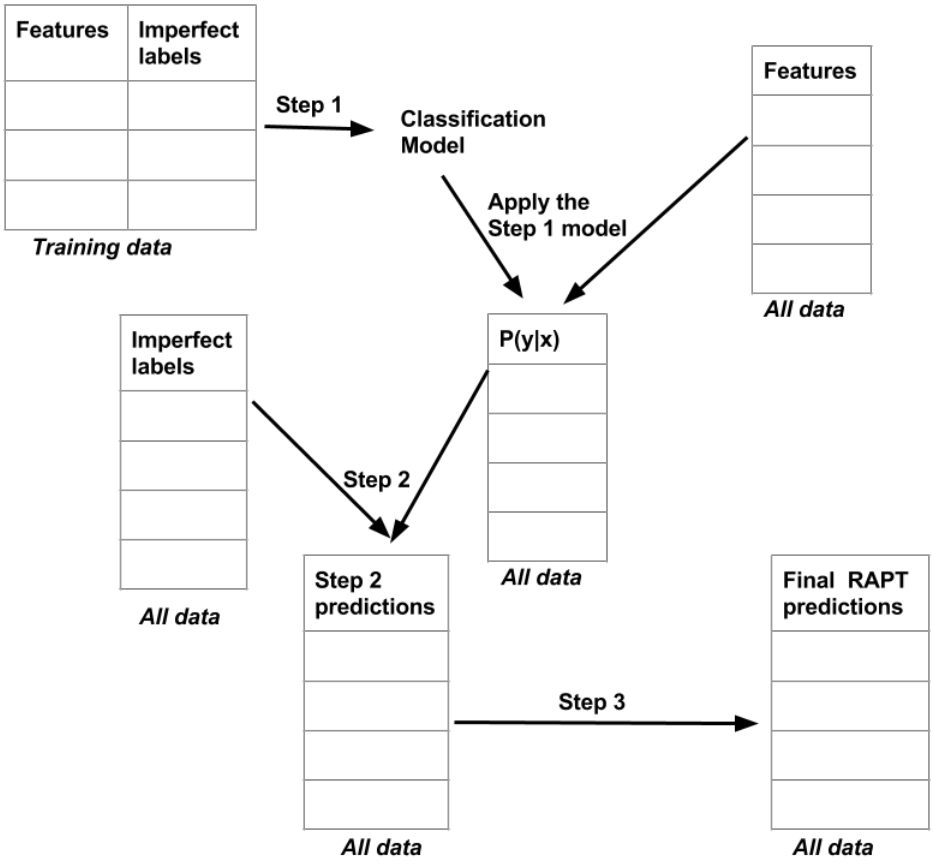



Figure 2.7: Flowchart representing the input and output data for each step of RAPT.

## 2.2 Background, Notations and Assumptions

In this section, we provide the traditional classification approach for rare classes, and the notations and assumptions used in the RAPT framework.

**Traditional Classification Approach**

In the traditional rare class classification setting, the learning algorithm is given a training set $D$ consisting of $n$ i.i.d. samples $\{(\mathbf{x}, y)\}_{i=1}^{n}$, where $\mathbf{x} \in X$ are explanatory variables and $y \in \{0, 1\}$ is target label. The learning algorithm selects a function $f$ from a function family $H$ that contain functions powerful enough to represent approximations of $P(y = 1|\mathbf{x})$ (or some monotonic function of $P(y = 1|\mathbf{x})$). A classifier $(f, \gamma)$ is written in the following form: $y = 1$ if $f(\mathbf{x}) \geq \gamma$ and 0 otherwise. The threshold $\gamma$ determines the operating point of the classifier. As $\gamma$ is swept from 0 to 1, recall decreases from 1 to 0,

while precision goes from a low value to a higher value. In practical settings, an optimal decision threshold, $\gamma_o^{f,y}$, is selected such that it jointly maximizes both precision and recall by optimizing measures such as harmonic mean (F-measure) or geometric mean (G-measure). The selection of threshold is done using a validation set (i.e., a subset of labeled samples that are representative of the test data distribution and have not been used for building the model) [16–18]. The approach presented is a modification to this basic procedure when only imperfectly labeled training samples are available.

**Assumptions**

Our approach makes use of the following assumptions:

*Assumption 1:* class conditional random label noise (CCN) [6, 7] This assumption implies that the probability of flipping the label of a given instance is independent of the attribute value $\mathbf{x}$ and depends only on the true class of the instance (i.e. $P(a|y, \mathbf{x}) = P(a|y)$). In fact, the CCN assumption is equivalent to the assumption that imperfect labels $a$ are independent of attributes $\mathbf{x}$ given true label $y$.

*Assumption 2:* $\alpha + \beta < 1$ [6, 7] This assumption is a requirement on the quality of the imperfect labels- the imperfect labels should be better than random (random annotations correspond to the condition $\alpha + \beta = 1$). This is only a mild requirement on quality of imperfect labels. Note that in previous studies it has been shown that as the sum of $\alpha$ and $\beta$ approaches 1, the requirement on the number of training samples needed increases considerably [6, 15]. Also note that this assumption is equivalent to the following condition: $P(a = 1|y = 1) > P(a = 1|y = 0)$ and $P(a = 0|y = 0) > P(a = 0|y = 1)$. These conditions simply mean that for any instance, the imperfect label $a$ is more likely to be 1 (or 0) if the true label for that instance is 1 (or 0).

*Assumption 3:* A sufficient number of training samples $x_s$ (eg. 5% of total data) are *perfectly negative*, i.e. $P(y = 1|x_s) = 0$. Similarly, a sufficient number of training samples $x_p$ (eg. 1% of total data) are *perfectly positive*, i.e. $P(y = 0|x_p) = 0$.

## 2.3 Step 1: Training classifier using imperfectly-annotated samples

In this section, we present a learning algorithm to build a classifier that maximizes the G-measure using imperfectly labeled training samples. Since we do not have access

to the true labels $y$, which are a necessary input for traditional supervised learning algorithms, we cannot directly learn the classifier $(f, \gamma_o^{f,y})$. In our approach, we make use of imperfect training samples $\{(\mathbf{x}, a)\}_{i=1}^n$, to first learn a function $g \in H$ that models $P(a = 1|\mathbf{x})$, and then select a threshold $\gamma_o^{g,y}$, such that performance of classifier $(g, \gamma_o^{g,y})$ is expected to be comparable to the desired classifier $(f, \gamma_o^{f,y})$.

Under the assumptions 1 and 2, it has been shown [6] that *the target function $f(\boldsymbol{x})$, is a monotonically increasing linear function of $g(\boldsymbol{x})$*. Hence, for every $\gamma_o^{f,y}$ on $f$, there exists a corresponding threshold $\gamma_o^{g,y}$ on $g$, such that performance of classifier $(g, \gamma_o^{g,y})$ is comparable to traditional classifier $(f, \gamma_o^{f,y})$ [6].

*Lemma:* $P(y = 1|\mathbf{x})$ is a monotonically increasing linear function of $P(a = 1|\mathbf{x})$ under assumptions 1 and 2.

*Proof*

$P(a = 1|\mathbf{x})$

$= 1 - P(a = 0|\mathbf{x})$

$= 1 - P(a = 0 \cap y = 0|\mathbf{x}) - P(a = 0 \cap y = 1|\mathbf{x})$

*Applying chain rule we get*

$= 1 - P(a = 0|y = 0, \mathbf{x})P(y = 0|\mathbf{x}) - P(a = 0|y = 1, \mathbf{x})P(y = 1|\mathbf{x})$

*Assuming $a$ is independent of $\boldsymbol{x}$ given $y$ we get*

$= 1 - P(a = 0|y = 0)P(y = 0|\mathbf{x}) - P(a = 0|y = 1)P(y = 1|\mathbf{x})$

$= 1 - P(a = 0|y = 0)(1 - P(y = 1|\mathbf{x})) - P(a = 0|y = 1)P(y = 1|\mathbf{x})$

Hence, $P(y = 1|\mathbf{x})$ can be written as a linear function of $P(a = 1|\mathbf{x})$:

$$P(y = 1|\mathbf{x}) = \frac{P(a=0|y=0)-1}{P(a=0|y=0)-P(a=0|y=1)} + \frac{P(a=1|\mathbf{x})}{P(a=0|y=0)-P(a=0|y=1)}$$

Further simplifying:

$$P(y = 1|\mathbf{x}) = \frac{P(a=1|\mathbf{x})-\beta}{1-(\alpha+\beta)}$$

As a result of the above relationship, if a training algorithm learns two functions $f(\mathbf{x})$ and $g(\mathbf{x})$ that model $P(y = 1|\mathbf{x})$ and $P(a = 1|\mathbf{x})$ (or some monotonic function of these probabilities), then

$$f(\mathbf{x}) = \frac{g(\mathbf{x})-\beta}{1-(\alpha+\beta)}$$

Note that perfect linear relationship exists only if $f(\mathbf{x})$ is an exact monotonic function of $P(y = 1|\mathbf{x})$ and $g(\mathbf{x})$ is an exact monotonic function of $P(a = 1|\mathbf{x})$. However, in practice this equality holds true only approximately because $f$ and $g$ are trained on finite training samples, and/or the family of models which $g$ is selected from may not include the true model. Hence, empirically the rankings are expected to be similar and not necessarily identical.

If true labels $y$ were available for a sample of instances, then the optimal decision threshold $\gamma_o^{g,y}$ that maximizes any desired performance measure (eg. F-measure or G-measure) can be easily selected. But the true labels $y$ are not available in our problem setting. In the following, we present a method to select optimal decision threshold $\gamma_o^{g,y}$ using only the imperfectly labels $a$ under an additional assumption.

First, we derive the expressions for the precision and recall of a classifier $(g, \gamma)$ with $y$ as target in terms of $a$, $\alpha$, $\beta$ and $P(y = 1)$. In particular, we prove that, under the assumptions 1 and 2, the precision ($Pre_\gamma^{g,y}$) and recall ($Rec_\gamma^{g,y}$) can be expressed as

$$Pre_\gamma^{g,y} = \frac{P(a=1|g(\mathbf{x})>\gamma)-\beta}{1-(\alpha+\beta)}$$
$$Rec_\gamma^{g,y} = \frac{[P(a=1|g(\mathbf{x})>\gamma)-\beta]P(g(\mathbf{x})>\gamma)}{[1-(\alpha+\beta)]P(y=1)}$$

**Expressing precision of classifier $(g, \gamma)$ in terms of $a$, $\alpha$, $\beta$ and $P(y = 1)$**

$P(a = 1|g(\mathbf{x}) > \gamma)$

$= \frac{P(a=1 \cap g(\mathbf{x})>\gamma)}{P(g(\mathbf{x})>\gamma)}$

*Applying chain rule we get*

$= \frac{P(a=1 \cap g(\mathbf{x})>\gamma|y=1)P(y=1)+P(a=1 \cap g(\mathbf{x})>\gamma|y=0)P(y=0)}{P(g(\mathbf{x})>\gamma)}$

*Assuming a is independent of x given y we get*

$= \frac{P(a=1|y=1)P(g(\mathbf{x})>\gamma|y=1)P(y=1)+P(a=1|y=0)P(g(\mathbf{x})>\gamma|y=0)P(y=0)}{P(g(\mathbf{x})>\gamma)}$

*Using Bayes rule we get*

$= P(a = 1|y = 1)P(y = 1|g(\mathbf{x}) > \gamma) + P(a = 1|y = 0)P(y = 0|g(\mathbf{x}) > \gamma)$

*Substituting* $P(y = 0|g(\boldsymbol{x}) > \gamma)$ *as 1 -* $P(y = 1|g(\boldsymbol{x}) > \gamma)$

$= P(y = 1|g(\mathbf{x}) > \gamma)[P(a = 1|y = 1) - P(a = 1|y = 0)] + P(a = 1|y = 0)$

Hence, $P(y = 1|g(\mathbf{x}) > \gamma)$ can be written as:

$$P(y = 1|g(\mathbf{x}) > \gamma) = \frac{P(a=1|g(\mathbf{x})>\gamma)-P(a=1|y=0)}{P(a=1|y=1)-P(a=1|y=0)}$$

Expressing in terms of flip probabilties:

$$Pre_{\gamma}^{g,y} = P(y = 1|g(\mathbf{x}) > \gamma) = \frac{P(a=1|g(\mathbf{x})>\gamma)-\beta}{1-(\alpha+\beta)}$$

**Expressing recall of classifier $(g, \gamma)$ in terms of $a$, $\alpha$, $\beta$ and $P(y = 1)$** $P(g(\mathbf{x}) > \gamma|y = 1)$

*Applying Bayes rule we get*

$= \frac{P(y=1|g(\mathbf{x})>\gamma)P(g(\mathbf{x})>\gamma)}{P(y=1)}$

*Using Building block 2 for estimating $P(y = 1|g(\boldsymbol{x}) > \gamma)$ we get*

$= \frac{[P(a=1|g(\mathbf{x})>\gamma)-P(a=1|y=0)]P(g(\mathbf{x})>\gamma)}{[P(a=1|y=1)-P(a=1|y=0)]P(y=1)}$

Expressing in terms of flip probabilities and skew:

$$Rec_{\gamma}^{g,y} = P(g(\mathbf{x}) > \gamma|y = 1) = \frac{[P(a=1|g(\mathbf{x})>\gamma)-\beta]P(g(\mathbf{x})>\gamma)}{[1-(\alpha+\beta)]P(y=1)}$$

Using these expressions for precision and recall, we can express the G-measure for a classifier $(g, \gamma)$ as:

$$(GM_{\gamma}^{g,y})^2 = Pre_{\gamma}^{g,y} \times Rec_{\gamma}^{g,y}$$
$$= \frac{[P(a = 1|g(\mathbf{x}) > \gamma) - \beta]^2 P(g(\mathbf{x}) > \gamma)}{[1 - (\alpha + \beta)]^2 P(y = 1)}$$

Thus, the threshold $\gamma_o^{g,y}$ is:

$$\gamma_o^{g,y} = \arg\max_{\gamma} \frac{[P(a=1|g(\mathbf{x})>\gamma)-\beta]^2 P(g(\mathbf{x})>\gamma)}{[1-(\alpha+\beta)]^2 P(y=1)}$$

Since the denominator is a constant that does not depend on $\gamma$, we only need to maximize the following objective:

$$\gamma_o^{g,y} = \arg\max_{\gamma}[P(a = 1|g(\mathbf{x}) > \gamma) - \beta]^2 P(g(\mathbf{x}) > \gamma)$$

Optimizing the above objective requires estimation of (i)$\hat{P}(g(\mathbf{x}) > \gamma)$, (ii) $\hat{P}(a = 1|g(\mathbf{x}) > \gamma)$, and (iii) $\hat{\beta}$. The terms (i) $\hat{P}(g(\mathbf{x}) > \gamma)$ and (ii) $\hat{P}(a = 1|g(\mathbf{x}) > \gamma)$ can be easily estimated using only imperfectly labeled $D_{corr}$ as follows:

$$\hat{P}(g(\mathbf{x}) > \gamma) = \frac{\sum_{i=1}^{n} \mathbb{1}(g(\mathbf{x}_i)>\gamma)}{n}$$

$$\hat{P}(a = 1|g(\mathbf{x}) > \gamma) = \frac{\sum_{i=1}^{n} \mathbb{1}(a_i=1 \cap g(\mathbf{x}_i)>\gamma)}{\sum_{i=1}^{n} \mathbb{1}(g(\mathbf{x}_i)>\gamma)}$$

Note that in case of PU learning settings [20, 22], $\beta$ is known to be 0. Hence, the optimization objective for threshold $\gamma_o^{g,y}$ becomes equivalent to the optimization objective of the method that treats imperfect labels as true labels (i.e., ignores corruption in

$a$). Thus, for PU learning settings G-measure becomes immune to label noise under the CCN assumption. However, in our problem setting $\beta$ cannot be assumed to be 0, and therefore needs to be estimated.

The straightforward method for estimating $\beta$ requires knowledge of the true labels $y$,

$$\hat{\beta} = \hat{P}(a = 1|y = 0) = \frac{\sum_{i=1}^{n} \mathbb{1}(a_i=1 \cap y_i=0)}{\sum_{i=1}^{n} \mathbb{1}(y_i=0)}.$$

Next, we show that for rare class scenarios, under the assumption that $P(y = 1|x) = 0$ for a sufficient number of training samples, $\beta$ can be estimated using imperfect labels of $D_{corr}$.

First, let us assume that there exists a *perfectly negative* sample $\mathbf{x}_n$ that has $P(y = 1|x_n) = 0$. It can be seen that

$$P(a = 1|\mathbf{x}_n) = \beta + (1 - (\alpha + \beta))P(y = 1|\mathbf{x}_n).$$

Under the assumption that $P(y = 1|x_n) = 0$, this implies that $\beta = P(a = 1|\mathbf{x}_n)$. The linear relationship between $P(y = 1|x)$ and $P(a = 1|x)$ helps in identifying $x_n$, as it will be assigned the minimum value of $P(a = 1|x)$(and hence also $g(x)$). These results gives us one way to estimate $\beta$ by using:

$$\hat{\beta} = min_{i=1}^{n} g(x_i).$$

However, the above estimate may have a high variance due to the impact of outliers in the samples [6, 20]. Thus, a more robust estimate is obtained under the assumption that a sufficient number of training samples $x_s$ (eg. 5% of total data) are *perfectly negative*, i.e. $P(y = 1|x_s) = 0$. Therefore, in our approach we select the bottom 5% of total instances sorted by $g(\mathbf{x})$, and then to estimate $\beta$ we compute $P(a = 1|\mathbf{x}_s)$ as the fraction of these instances that are assigned positive by imperfect labels. Under the assumption that $P(y = 1|x) = 0$ for these samples, the estimate of $P(a = 1|\mathbf{x}_s)$ gives the correct estimate for $\beta$ that is also robust to presence of outliers in training samples.

The above estimate of $\beta$ will be biased if there is presence of positive samples in the bottom 5% of instances (sorted by $g$). Positive samples may be present in the bottom 5% instances if (i) the assumption that $P(y = 1|x) = 0$ is violated at 5% of total samples, or (ii) $g$ is not a good model for $P(a = 1|x)$. The presence of positive samples

will lead to an over-estimation of $\beta$. Note that the overestimation in $\hat{\beta}$ decreases as the imbalance between classes is increased (i.e. $P(y = 1)$ becomes close to 0). This is because for any scoring function $g$ that has a performance better than random ordering, the $P(y = 1|x)$ in the bottom 5% cannot exceed $P(y = 1)$. This property implies that the estimates of $\beta$ have a lower bias in rare class problem setting.

Note that performance of the classifier $(g, \gamma_o^{g,y})$ can only be expected to approach the performance of classifier $(f, \gamma_o^{f,y})$. Thus, if $(f, \gamma_o^{f,y})$ show poor performance on some data set, either due to overlap between the classes in the feature space or the inability of functions of $H$ to model the relationship between features and target, then performance of $(g, \gamma_o^{g,y})$ will also be poor.

## 2.4 Step 2: Combining multiple predictions

In this step, we present a strategy to improve the performance of rare class predictions by combining information from (i) the imperfect labels and (ii) the predictions from function $g$ trained in the first step. Note that to use this step it is necessary that imperfect labels are available for all instances (characteristic 2).

Previous work in machine learning [24–26] has studied the problem of combining information present in multiple imperfect annotations to produce a *more accurate* prediction. These methods assume that there is an underlying unobserved true label and the observed imperfect annotations are perturbations of this true label. The divergence between the annotator labels and the true label depends on the annotator quality. A commonly used approach is to model the annotator quality and true labels as latent variables and use a joint optimization framework to infer them by maximizing the likelihood of observed imperfect annotations.

However, this approach is not applicable in our current setting due to the following two reasons. First, its efficacy depends on the availability of several predictors, while we have access to only two sets of predictions- (i) the imperfect labels and (ii) the predictions from classification model trained in the first step. The second difference is that the previous algorithms for combining multiple predictions maximize classification accuracy, while the focus of this step is a combination strategy that jointly maximizes the precision and recall of the final prediction.

Step 2 of RAPT uses a simple combination step that takes the *logical AND* of the two sets of predictions to produce the combination output. More specifically, given the two sets of prediction sources for all test instances - one from imperfect labels $a$ and one from function $g$ trained in step 1, the combination step of RAPT gives a prediction $c_i$ for each instance $i$ based on a decision threshold on the function $q(x) = a \times g(x)$ as follows

$$\begin{aligned} c_i &= 1 && \text{if } q(x) > \gamma \\ &= 0 && \text{otherwise} \end{aligned}$$

It is obvious that the rare class detection output from such a combination step will be more conservative than both individual predictors- imperfect labels $a$ and classifier $(g(\mathbf{x}), \gamma)$. Thus, the combined output $c$ will have a lower false positive rate (FPR) than the two individual predictors, and hence higher precision of the rare class output compared to individual predictors. Moreover, due to the conservative nature of the combined output $c$, it will have a lower recall compared to individual predictors.

We present a key result for combination step, which shows that in context of rare class scenarios, the gain in precision can be much higher than the loss in recall, and thus results in an improvement in the G-measure compared to the classifier $(g, \gamma)$.

We also present a method to select a threshold $(\gamma_o^{g,y})$ on $q(\mathbf{x})$ such that this threshold maximizes the G-measure at the end of step 2. This new threshold $\gamma_o^{g,y}$ is expected to be lower than the threshold selected in step 1, as the combination step increases precision and decreases recall. Moreover, even though in general the step 2 is expected to increase precision as the expense of decreasing recall with respect to step 1, sometimes, as a consequence of this threshold adjustment, the combination step may increase both recall and precision.

### 2.4.1 Gain after combination step

To compare the G-measure after step 2 with the G-measure of step 1, we start with the expression for product of precision and recall of a classifier $(g, \gamma)$ derived above.

$$Pre_\gamma^{g,y} \times Re_\gamma^{g,y} = \frac{[P(a = 1|g(\mathbf{x}) > \gamma) - \beta]^2 P(g(\mathbf{x}) > \gamma)}{[1 - (\alpha + \beta)]^2 P(y = 1)}$$

Similarly, the expression for product of precision and recall of combination output is given as:

$$\frac{[P(y=1|c=1)]^2 P(c=1)}{P(y=1)}$$

Under the CCN assumption $x$ is independent of $a$ given $y$. This also implies that the function $g(\mathbf{x}) > \gamma$ is independent of imperfect label $a$ given $y$.

**Precision of classifier ($c$) for each candidate threshold $\gamma$ can be written as following**

$$P(y=1|c=1)$$

*using Bayes rule*

$$= \frac{P(c=1|y=1)P(y=1)}{P(c=1)}$$

*substituting $c$ as $a = 1 \cap g(\boldsymbol{x}) > \gamma$*

$$= \frac{P(a=1 \cap g(\mathbf{x}) > \gamma|y=1)P(y=1)}{P(c=1)}$$

*using independence between $a$ and $\boldsymbol{x}$ given $y$*

$$= \frac{P(a=1|y=1)P(g(\mathbf{x}) > \gamma|y=1)P(y=1)}{P(c=1)}$$

*using Bayes rule*

$$= \frac{P(a=1|y=1)P(y=1|g(\mathbf{x}) > \gamma)P(g(\mathbf{x}) > \gamma)P(y=1)}{P(y=1)P(c=1)}$$

*using building block 2*

$$= \frac{P(a=1|y=1)[P(a=1|g(\mathbf{x}) > \gamma) - P(a=1|y=0)]P(g(\mathbf{x}) > \gamma)}{P(c=1)[P(a=1|y=1) - P(a=1|y=0)]}$$

*separating terms that depend on $\gamma$*

$$= \frac{[P(a=1|g(\mathbf{x}) > \gamma) - P(a=1|y=0)]P(g(\mathbf{x}) > \gamma)P(a=1|y=1)}{P(c=1)[P(a=1|y=1) - P(a=1|y=0)]}$$

*substituting $c$ as $a = 1 \cap g(\boldsymbol{x}) > \gamma$*

$$= \frac{[P(a=1|g(\mathbf{x}) > \gamma) - P(a=1|y=0)]P(a=1|y=1)}{P(a=1|g(\mathbf{x}) > \gamma)[P(a=1|y=1) - P(a=1|y=0)]}$$

Substituting the above expression for precision:

$$\frac{[\frac{[P(a=1|g(\mathbf{x})>\gamma)-P(a=1|y=0)]}{P(a=1|g(\mathbf{x})>\gamma)}\frac{P(a=1|y=1)}{[P(a=1|y=1)-P(a=1|y=0)]}]^2 P(c=1)}{P(y=1)}$$

Using these expressions for the product of precision and recall corresponding to step 1 and step 2 of RAPT, the gain factor in G-measure after Step 2 can be written as:

$$\frac{GM(step2)}{GM(step1)} = \frac{1-\alpha}{\sqrt{P(a=1|g(\mathbf{x})>\gamma)}}$$

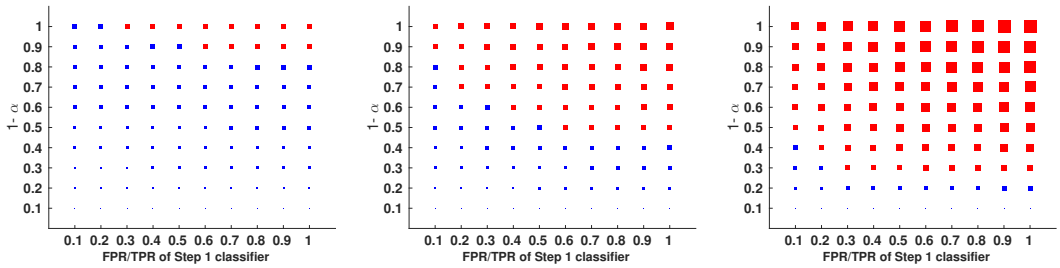
$P(a = 1|g(\mathbf{x}) > \gamma)$ is the estimated precision according to imperfect labels $a$. It decreases when the overlap between the classes in the feature space is increased or the imbalance between the classes is increased. A higher gain in G-measure after step 2 is observed for scenarios with low values of estimated precision, which is exactly where step 2 is needed. Furthermore, a high value of $\alpha$ leads to more positive instances being incorrectly assigned to negative class after step 2 even though they are correctly classified in step 1. Thus, for high values of $\alpha$, there may be no gain in G-measure, as the reduction in recall may exceed the increase in precision.

Figure 2.8 illustrates how the gain after step 2 increases with (i) $1-\alpha$, (ii) FPR/TPR of classifier on $\mathbf{x}$, and (iii) skew between classes. Figure 2.8(a) shows the gain corresponding to balanced classes (skew 1:1). We notice that there is only a small gain that too under conditions of extremely small values of $\alpha$ and a very high overlap (FPR/TPR) between classes. Thus, the combination step has little utility for balanced classes. In contrast, as the skew is increased to 1:10 (see Figure 2.8(b)), we observe that a considerable gain (upto a factor of 5) is obtained across many combinations. Finally, for a high skew such as 1:100 we observe that there are significant gains (upto a factor of 10) in G-measure for most combinations (see figure 2.8(c)).

### 2.4.2 Fixing threshold to directly optimize G-measure of combination output

We maximize the G-measure of the combination step output by selecting a new threshold $\gamma_o^{q,y}$ on $q$ as follows:

$$\gamma_o^{q,y} = \arg\max_\gamma \frac{[P(y=1|c=1)]^2 P(c=1)}{P(y=1)}$$




(a) Skew = 1:1 (b) Skew = 1:10 (c) Skew = 1:100

Figure 2.8: Gain factor in G-measure after step 2 for different values of $\alpha$, overlap between classes (measured as the ratio FPR/TPR) and skew between classes. The x-axis corresponds to the overlap between classes in the feature space measured as the ratio FPR/TPR, while the y-axis corresponds to the value of $1-\alpha$. Each *square* corresponds to the gain in G-measure after step 2 for a combination of $\alpha$ and FPR/TPR. The size of the *square* indicates the magnitude of the gain. For ease of visualization, the *squares* with value less than 1 are colored in *blue*, while the *squares* with value greater than 1 are colored in *red*.

Since $P(y=1)$ is a constant not depending on $\gamma$ we get:

$$\gamma_o^{q,y} = \arg\max_\gamma [P(y=1|c=1)]^2 P(c=1)$$

Using the equation for precision of combination step output (and removing the constant terms) gives

$$\gamma_o^{q,y} = \arg\max_\gamma [\frac{[P(a=1|g(\mathbf{x})>\gamma) - P(a=1|y=0)]}{P(a=1|g(\mathbf{x})>\gamma)}]^2 P(c=1)$$

Earlier, we provided a method to compute $\hat{P}(a=1|g(\mathbf{x})>\gamma)$ and $\hat{P}(a=1|y=0)$. Similarly, $\hat{P}(c=1)$ is estimated from data as the fraction of instances with $a=1$ and $g(\mathbf{x})>\gamma$.

Using the above results, we estimate geometric mean of precision and recall on the output of second step (upto a proportionality constant) corresponding to different values of threshold $\gamma$ and select the value of $\gamma$ that maximizes the estimated geometric mean. This threshold $\gamma_o^{q,y}$ is used to get binary predictions from classifier $(q, \gamma_o^{q,y})$.

## 2.5 Step 3: Collective classification step

The combination step of RAPT provides a set of confident positive (rare) class instances, and also a classifier $(g, \gamma_o^{g,y})$ that optimizes the G-measure of step 2. The predictions of step 2 incur a false negative whenever either the classifier $(g, \gamma_o^{g,y})$ or imperfect label $a$ assigns a positive instance to the negative class. Thus, the step 2 output may have a low recall, especially if the recall of imperfect labels being used is low. The collective classification step of RAPT leverages the "linked" behavior of instances to reduce the number of false negatives (and hence increase the rare class recall). In particular, instances that are assigned to positive class according to classifier $(g, \gamma_o^{g,y})$ but excluded by the combination step due to imperfections in $a$ (i.e. a non-zero $\alpha$ of the imperfect labels) are included in the final rare class prediction if they are spatially close to some of the *confident* positive instances identified in the combination step.

The collective classification step used in RAPT assigns instances to the positive class based on a spatially smooth density of step 2 output $(\pi)$ and the predictions from the classifier $(g, \gamma_o^{g,y})$. Specifically, the step 2 density at pixel $i$ takes high values if either the node $i$ or its nearby nodes are labeled positive by the step 2. We used a simple definition for the spatial density given as $\pi_i = \max(c_i, \frac{\sum_{j \in n(i)} e^{-(1-c_j)}}{|n(i)|})$, where $c_i$ is the step 2 output and $n(i)$ are the spatial neighbors of instance $i$. An instance is assigned to the positive class in step 3 when $\pi_i > 0.5$ and $g(x_i) > \gamma_o^{g,y}$.

Note that all instances that are labeled as positive by step 2 of RAPT will satisfy the above condition. However, some instances that are labeled as negative class in step 2 get assigned to positive class in this step, if they are assigned to positive class by the step 2 classifier $(g, \gamma_o^{g,y})$ and they are in spatial proximity of other confident positives that were identified in step 2. Negative instances that are incorrectly assigned to positive class by $(g, \gamma_o^{g,y})$ are typically not included in positive set by this step, as their $\pi$ value is generally quite low, thus maintaining a high precision of the final rare class output.

Step 3 can be applied as long as there is some notion of neighborhood. In the applications illustrated here, the concept of spatial neighborhood was natural. In other applications (e.g., social network graphs) neighborhood may be defined using path length.

## 2.6 Real-world Applications

We evaluate the RAPT framework on two real world environmental applications: mapping of forest fires and urban extent using satellite data. In this section we provide the background and details of data used for the forest fire and urban mapping applications and the evaluation setup for the two applications.

**Forest Fire Mapping:** For the forest fire mapping application, the RAPT framework uses two remotely sensed composite data products from the MODIS instrument aboard NASA's Terra satellite, which are available for public download [13]. Specifically, we use the Enhanced Vegetation Index (EVI) from the MODIS 16-day Level 3 1km Vegetation Indices (MOD13A2) and the Active Fire (AF) from the MODIS 8-day Level 3 1km Thermal Anomalies and Fire products (MOD14A2). EVI essentially measures "greenness" (area-averaged canopy photosynthetic capacity) as a proxy for the amount of vegetated biomass at a particular location. We extracted *vegetation difference* feature from vegetation profile (EVI) of locations that can be used to distinguish between the burned and unburned pixels [28]. A logistic regression classifier is built on the *vegetation difference* feature to predict the probability of target class. MODIS Active Fire (AF) product is used as the heuristic that provides the "imperfect labels". AF is *true* if a severe temperature anomaly is observed at a pixel on a given time step and *false* otherwise. Burned pixels are more likely to have an Active Fire signal on the date of fire compared to other unburned pixels. However, due to uncertainty in data collection and interference due to clouds and smoke, there are both false positives and false negatives if AF is used as a surrogate for burning activity [10]. For example, in the 4 data sets used in this study the precision of AF varies between 0.65 - 0.80 and its recall varies between 0.65 - 0.93.

**Urban Extent Mapping:** The urban area mapping application uses the multispectral data (MOD09A1) available at 500m spatial resolution at 16-day frequency and the Night-time light data, which is available annually [13]. Multispectral data captures the reflectance signal from land surface and has been used to distinguish urban land cover from other land classes [29]. A Support Vector Machine is built on the 7-dimensional spectral band feature space to predict the target class. Night-time light data captures the intensity of light radiated from each pixel at night and can serve as an imperfect

label for urban settlements globally. The night-time light data is often highly diffused and/or misregistered because of which several non-urban pixels in spatial neighborhood of urban settlements tend to also show a high intensity.

**Evaluation setup:** We evaluate our results for forest fire monitoring in the states of California, Georgia and Montana in U.S.A. for which we obtained fire validation data from government agencies responsible for monitoring and managing forests and wildfires (http://www.mtbs.gov/dataaccess.html). The validation data is in the form of fire polygons, each of which is associated with the time of burning. We consider an event to be positive if the corresponding pixel lies completely inside a polygon. Similarly, an event is considered to be unburned (forming the negative class) only if the entire pixel is outside a polygon. Since it is difficult to decide the class (burned/unburned) for a pixel which is partially inside the polygons, pixels that partially overlap polygon boundaries are discarded from the evaluation framework to avoid ambiguity. Similarly, for urban application we evaluate RAPT in the cities of Rochester and Mexicalli for which we manually created urban perimeters using higher resolution Google Earth imagery. Note that although best efforts were made in documenting the fire and urban perimeter datasets, they are neither complete nor without error due to finite resources available.

## 2.7 Experimental Results

We experimentally evaluated the RAPT framework on datasets from two real-world applications.

### 2.7.1 Evaluation of the three steps of RAPT framework

We evaluated the three steps of the RAPT framework on 4 datasets for burned area application (states of Montana, Georgia, North and South California) and 2 datasets for urban mapping application (cities of Mexicali and Rochester). Figures 2.9 and 2.10 show the precision and recall corresponding to the classifier trained on gold standard labels (*as inverted triangle*), RAPT step 1 classifier trained on imperfect labels (*as triangle*), RAPT step 2 output (*as circle*), RAPT step 3 output (*as diamond*), and imperfect labels used as input in the framework (*as square*) for all 6 datasets.

**Comparable performance of the step 1 classifiers-** $(g, \gamma_o^{g,y})$ **and** $(f, \gamma_o^{f,y})$

As expected from our theoretical analysis, the performance of the two classifiers is comparable in these datasets. In fact, we notice in Figures 2.9(a), 2.9(b), and 2.9(c) that the precision and recall of $(g, \gamma_o^{g,y})$ and $(f, \gamma_o^{f,y})$ is almost identical (i.e., the *triangle* and *inverted triangle* have almost identical precision and recall). Note that in all 6 datasets the precision and recall values of RAPT step 1 are very similar to each other. This is expected as we optimized G-measure, which avoids increasing precision at expense of recall or vice-versa.

**Big improvement in precision $\times$ recall after step 2** From Figures 2.9 and 2.10, we observe that there is an increase in precision after Step 2 for each test dataset. For example, Figure 2.9(a) shows that the precision increases drastically from 0.82 to 0.96 for the state of N. California after combination step. As expected, the increase in precision is also accompanied by a loss in recall in most cases. But we observe that the loss in recall is much smaller compared to the increase in precision, especially if the imperfect label has a very high individual recall. As an example, the loss in recall is only marginal in case of N. California (Figure 2.9(a)) and S. California (Figure 2.9(b)), which have a very high recall for Active Fire. On the other hand, in case of Georgia (Figure 2.9(d)) that has a low recall for Active Fire, there is a considerable loss in recall after Step 2. Furthermore, in case of Montana (see Figure 2.9(c)) recall improves after step 2, as AF has a high recall in this region and the combination step allows the threshold on $g$ to be lowered.

**Improvement in recall after step 3** In Figures 2.9 and 2.10 we observe that the *collective classification* step improves the recall of the rare class without significantly reducing its precision. As expected, the maximum gain in recall is observed in datasets where there was a considerable loss in recall after step 2 due to the poor recall of imperfect labels (eg. see results for Georgia in Figure 2.9(d)). The slight loss in precision is due to the inclusion of some false positives in the spatial neighborhood. However, for the datasets used in this study the gain in recall exceeded the loss in precision; thus improving the overall performance of rare class detection. In our evaluation we have used neighbors in a 5 X 5 spatial neighborhood for Step 3. The size of spatial neighborhood is a parameter than needs some tuning based on the specific application, as the number of neighbors used impact the results. For instance, if we keep the spatial neighborhood to be too small, then the improvement in recall observed in Step 3 of

RAPT will be much smaller. On the other hand, if we considerably increase the size of the spatial neighborhood, then precision could decrease in cases where a large region is incorrectly labeled as belonging to the positive class by Step 1 and because of its large size, it happens to have at least one location labeled as positive by the imperfect label by random chance. In our applications, we found that using spatial neighborhood in the range 5X5 to 20X20 achieved the desired improvement in recall after Step 3 without a significant loss in precision.

**Comparison of RAPT predictions with imperfect labels used to train RAPT framework** The imperfect labels used (Active fire and Nighttime lights) can vary between 0.60 to 0.95 in their precision and recall values (as seen in Figures 2.9 and 2.10). Our results show that the performance of RAPT step 3 output (shown as *diamond*) is considerably better compared to imperfect labels (shown as *square*) for all 6 data sets evaluated in this study. In fact, the precision and recall values for RAPT step 3 are between 0.90 - 0.95 for all the 6 datasets evaluated in this study. The RAPT framework is able to achieve such high performance across all datasets because the two classes are distinguishable in the attribute space $\mathbf{x}$ (i.e. vegetation difference and spectral bands) and RAPT was successful in building classifiers using imperfect labels.

### 2.7.2 Estimating G-measure and selecting the optimal threshold using imperfect labels as target

In RAPT step 1 we presented a method to select the optimal threshold that maximizes G-measure with $y$ as target using only an imperfectly-labeled validation set. Our method is expected to give the correct threshold (i.e. in accordance with true labels) because it can estimate the product of precision and recall (upto a proportionality constant). Figure 2.11 shows the comparison of (scaled) estimated G-measure using RAPT (*in blue*) and G-measure computed using true labels (*in red*) corresponding to different thresholds. The figures clearly show that the estimated G-measure (using RAPT on imperfectly-labeled validation set) has same trend as the true G-measure and hence can be used to select the optimal threshold.

### 2.7.3 Comparison of RAPT Step 1 method with naive method (that ignores label noise)

As illustrated by Figure 2.4, the performance of the naive approach degrades as the probability of flipping negative labels ($\beta$) is increased. Specifically, we created multiple CCN corrupted datasets, each with different $\beta$ of input imperfect labels, by perturbing the true labels in the California state dataset. We observed that the performance of naive approach (in *blue*) decreases sharply compared to the ground truth labels-based learning (in *red*) when the $\beta$ of the imperfect labels is increased. Figure 2.12 shows the performance of classifiers built on imperfect labels using the RAPT step 1 method (in *black*) along with the *red* and *blue* plots from Figure 2.4. We observe that the RAPT Step 1 classifier is quite robust to perturbations of labels and the performance of the RAPT classifier (in *black*) is similar to that of ground truth-based classifier till $\beta < 0.6$. The value of $\alpha$ for this data set was 0.4, therefore when $\beta$ exceeded 0.6, the assumption $\alpha + \beta < 1$ gets violated resulting in extremely poor performance. This experiment demonstrates that the RAPT method for fixing threshold using the imperfectly-labeled validation set is quite robust. Note that in this experiment we only reported the divergence in performance of RAPT classifier and naive classifier when $\beta$ is increased (keeping $\alpha$ fixed). A similar divergence in performance is not observed when $\alpha$ is increased (keeping $\beta$ fixed). This asymmetry in the impact of $\alpha$ and $\beta$ on performance of naive method can be understood from our results in the Section 3 where we show that in presence of imperfection in training labels one should select threshold that maximizes:

$$\gamma_o^{g,y} = \arg\max_\gamma [\hat{P}(a = 1|g(\mathbf{x}) > \gamma) - \hat{\beta}]^2 \hat{P}(g(\mathbf{x}) > \gamma)$$

while the naive method selects threshold based on :

$$\gamma_o^{g,a} = \arg\max_\gamma [\hat{P}(a = 1|g(\mathbf{x}) > \gamma)]^2 \hat{P}(g(\mathbf{x}) > \gamma)$$

Since these two objectives differ only in the term $\hat{\beta}$, the naive method selects incorrect threshold if $\hat{\beta}$ is high (as seen in Figure 2.12).

### 2.7.4 Comparison of RAPT Step 1 method with other baseline methods

We compare the RAPT Step 1 classifier against two other baseline approaches to show the advantage of using the method proposed in Step 1 over related work. The first baseline method we consider is SMOTE [8], a resampling approach that over-samples the rare class using synthetic data generation. Though commonly used in rare class analysis, since this approach does not explicitly address the issue of presence of noise in the labels, Figure 2.13 clearly shows that the performance (as indicated by black curve) goes down when noise is added to the labels.

The second baseline method is a PU learning algorithm [30] that trains a classifier to optimize G-measure under the PU assumption, i.e. only positive labels are flipped. Since, PU learning does not account for flips in negative samples (i.e., $\beta$), Figure 2.13 shows that this algorithm is not robust to increase in probability of flipping negative samples (i.e., $\beta$). The best performance is obtained using RAPT step 1 method (blue curve), which is robust till the condition $\alpha + \beta < 1$ gets violated at $\beta = 0.6$.

## 2.8 Can we estimate model performance using imperfect labels?

RAPT Step 1 as well as the previous studies [6, 7] focus on building classification models using imperfect labels. In practical settings it is also desirable to get an estimate of the expected performance of any classifier before it is deployed. For example, in the forest fire monitoring application, it is desirable to know the expected precision and recall of the model trained in Step 1 on new test data.

Estimating model performance using imperfect labels alone is challenging because the precision and recall computed on imperfect labels $a$ do not match the desired precision and recall with respect to true labels $y$. However, Menon et al [6] have shown that for a very large class of performance measures (such as balanced error rate and area-under-curve), if we know $\alpha$ and $\beta$ then the model performance can be estimated using only the imperfect labels. Using a similar approach we can estimate both precision and recall from the imperfect labels $a$ alone.

$$Pre_{\gamma}^{g,y} = \frac{P(a=1|g(\mathbf{x})>\gamma)-\beta}{1-(\alpha+\beta)}$$
$$Rec_{\gamma}^{g,y} = \frac{[P(a=1|g(\mathbf{x})>\gamma)-\beta]P(g(\mathbf{x})>\gamma)}{P(y=1)(1-(\alpha+\beta))}$$

Earlier we discussed a way to estimate $\beta$. A similar strategy can be used to estimate $\alpha$. In particular, we estimate $\alpha$ as the fraction of negative imperfect labels in the top $k$ instances (eg. 1% of total instances) sorted according to $g(\mathbf{x})$. The choice of $k$ controls the bias and variance of this estimate; a small value of $k$ will have a lower bias, as the top few instances are more likely to satisfy the assumption that $P(y = 0|x) = 0$, but a small value of $k$ will result in a high variance of the estimate due to the impact of outliers in training samples. On the other hand, using a large value of $k$ will result in a high bias of the estimate, as it is unlikely that $P(y = 0|x)$ is 0 for all $k$ instances. Note that the correctness of the estimates of precision and recall would depend on the correctness of the estimates of $\alpha$ and $\beta$.

The robustness of the estimates of $\alpha$ and $\beta$ depends on the additional assumptions that the top 1% ($\mathbf{x}_P$) and the bottom 5% ($\mathbf{x}_N$) data instances have $P(y = 1|x_P) = 1$ and $P(y = 1|x_N) = 0$, respectively. In practice, these assumptions are violated due to (i) overlap in the feature space and (ii) inability of the hypothesis space to learn complex decision boundaries. The value of $P(y|x)$ will never be perfectly 0 or 1 if there exists an overlap between the classes in the feature space. Thus, the most positive sample will have a class probability less than 1, and the most negative sample will have a class probability greater than 0. The deviation of the class probability for the most positive (or negative) sample from 1 (or 0) depends on the extent of the overlap in the feature space. Furthermore, if the hypothesis space used to model $P(a|x)$ does not contain functions powerful enough to model $P(a|x)$, then the ranking of $g$ may not be identical to $P(a|x)$ (and hence to $P(y|x)$). As a result, the top 1% instances (or bottom 5% instances) sorted according to $g$ do not necessarily correspond to the the top 1% instances (or bottom 5% instances) according to $P(y|x)$. As a consequence of the violations of these assumptions, both $\alpha$ and $\beta$ tend to be overestimated in practice. However, the overestimation term in $\beta$ decreases as the imbalance between classes is increased, while for $\alpha$ it increases. Thus, for rare class problem settings, the estimates of $\beta$ are expected to be more robust than $\alpha$ (i.e. have lower bias).

We investigated the robustness of the estimates of $\alpha$ and $\beta$ on our data sets. As an example, Figure 2.14 shows the actual value of $\alpha$ and $\beta$ in Georgia data set, and the

mean and standard deviation of the estimates $\hat{\alpha}$ and $\hat{\beta}$ at different fraction of instances used for estimation. We observe that both $\alpha$ and $\beta$ are underestimated at extremely low fractions (i.e. when only a small number of instances are used for estimation). This is both due to the presence of outliers and violations of CCN at the positive and negative extremity. When sufficient number of instances are used for estimation, as expected, we observe that $\hat{\beta}$ is more robust to the choice of fraction of instances used for estimation, compared to $\hat{\alpha}$ that has a considerably higher bias and variance. Note that if it was possible to obtain robust estimates of both $\alpha$ and $\beta$, one can also estimate precision and recall, and hence can maximize other performance measures also, eg. the commonly used F-measure. It is because the estimates of $\alpha$ are not robust that we optimize G-measure in RAPT, since G-measure can be optimized using only $\hat{\beta}$.

| tile | Precision | Est. Precision | L.B. Precision | Recall | Est. Recall |
|---|---|---|---|---|---|
| N. California | 0.79 | 0.82 | 0.73 | 0.72 | 0.69 |
| S. California | 0.65 | 0.69 | 0.63 | 0.81 | 0.66 |
| Montana | 0.92 | 0.91 | 0.87 | 0.79 | 0.65 |
| Georgia | 0.63 | 0.78 | 0.50 | 0.57 | 0.69 |

Table 2.1: Table reports the actual precision and recall computed using $y$ as target, and estimated precision and recall computed for fire data sets. Table also reports the estimated lower bound on precision.

We used these estimates of $\hat{\alpha}$ and $\hat{\beta}$ to estimate precision and recall and compare them with the actual precision and recall for the 4 fire monitoring data sets (see Table 2.1). We notice that the estimates of precision and recall are not trustworthy, as expected given the divergence between the actual value of $\alpha$ and its estimated value.

Even though we are not able to estimate precision and recall accurately, it is possible to compute a lower bound on precision given $\hat{\beta}$ (which we saw can be robustly estimated under rare class settings). In particular, $\frac{P(a=1|g(\mathbf{x})>\gamma)-\hat{\beta}}{1-\hat{\beta}}$ can be used as a lower bound on precision $LB(Pre_\gamma^{g,y})$. This lower bound expression assumes that $\hat{\beta}$ is an overestimate of $\beta$. The third column in Table 2.1 reports the $LB(Pre_\gamma^{g,y})$ for each fire monitoring dataset. Note that the $LB(Pre_\gamma^{g,y})$ can be low either because (i) the actual precision $Pre_\gamma^{g,y}$ is low, or (ii) the value of $\alpha$ is high (e.g., if $\alpha$ is 0.5 and $\beta$ is 0, $LB(Pre_\gamma^{g,y})$ will be half of $Pre_\gamma^{g,y}$). Since optimizing G-measure ensures that both precision and recall

at the decision threshold are comparable, and one of them is not high at the expense of the other. Hence, the value of $LB(Pre_\gamma^{g,y})$ can be used to decide if the classification model meets a desired performance level, failing which it should not be deployed.

**Deriving $LB(Pre_\gamma^{g,y})$ lower bound on precision**

$$Pre_\gamma^{g,y} = \frac{P(a=1|g(\mathbf{x})>\gamma)-\beta}{1-(\alpha+\beta)}$$

$$\textit{Moreover, } \frac{P(a=1|g(\mathbf{x})>\gamma)-\beta}{1-(\alpha+\beta)} > \frac{P(a=1|g(\mathbf{x})>\gamma)-\hat{\beta}}{1-(\alpha+\hat{\beta})}$$

$$\textit{when}$$

$$(\hat{\beta}-\beta)(1-\alpha-P(a=1|g(\mathbf{x})>\gamma))>0$$

$$\textit{which is true since } \hat{\beta} \textit{ is expected to be an overestimate of } \beta$$

$$\textit{and } P(a=1|g(\boldsymbol{x})>\gamma) < 1-\alpha \textit{ under the}$$

$$\textit{constraint that } P(y=1|g(\boldsymbol{x})) \leq 1$$

$$\textit{Also since } \alpha \textit{ is a number between 0 and 1}$$

$$\textit{by substituting } \alpha \textit{ as 0, we get}$$

$$\frac{P(a=1|g(\mathbf{x})>\gamma)-\hat{\beta}}{1-(\alpha+\hat{\beta})} > \frac{P(a=1|g(\mathbf{x})>\gamma)-\hat{\beta}}{1-\hat{\beta}}$$

$$\textit{This gives a lower bound on } Pre_\gamma^{g,y}$$

$$\frac{P(a=1|g(\mathbf{x})>\gamma)-\hat{\beta}}{1-\hat{\beta}} = LB(Pre_\gamma^{g,y})$$

## 2.9 How to construct imperfect labels that satisfy CCN assumption?

We obtain the imperfect labels for the two applications discussed using Active Fire and Night-time light that are standard MODIS products [13], which are available for all locations. We believe that in many applications, it is possible for the domain experts to provide a heuristic on a single feature or a subset of features to construct imperfect labels, which can be used in RAPT framework. The class conditional random label noise (CCN) assumption on the imperfect labels requires that the features involved in generating the imperfect labels should be different from the explanatory variables (denoted as $\mathbf{x}$) on which the Step 1 classifier is trained. Therefore using domain-guided heuristics on features to obtain the imperfect labels for RAPT is feasible only when there are at least two different feature subsets in data such that attributes of both subsets can reasonably discriminate the two classes and are conditionally independent

on the class label of the instances.

In many real-world applications, the CCN assumption on imperfect labels may be violated due to the presence of *data heterogeneity*. For example, in earth science data the properties of the imperfect labels often vary due to a number of factors including geography, seasons and land cover. As an example, the probability of missing a fire event (i.e., $\alpha$) is higher for tropical regions compared to other regions (eg., California) due to the persistent cloud cover in tropics. Hence, the conditional independence between the imperfect labels and features is violated (greater probability of error in the feature subspace corresponding to the tropics compared to California). In our study we partitioned the earth science data into relatively homogeneous data subsets. These smaller homogeneous partitions are more likely to follow the CCN assumption. It is important to note that we can apply the RAPT framework for each partition independently using the imperfectly labeled samples from its own partition, since imperfect labels are available for all samples. Note that excessive partitioning of the data though increases homogeneity, but may often result in poor performance due to scarcity of positive class samples in some partitions.

## 2.10 Concluding remarks

Even though, the performance of RAPT has been evaluated in regions where ground truth is known (forest fires) or can be constructed manually (urbanization), the real utility of the paradigm is in areas where no ground truth is available. Indeed the RAPT framework has been used to produce the first comprehensive and high quality history of fires in the tropical forests in South America and South east Asia, where many of these fires are known to be a precursor to illegal conversions of tropical forests to plantations and other agricultural uses [4]. These results are also publicly available via a web viewer `http://z.umn.edu/fireviewer/`.

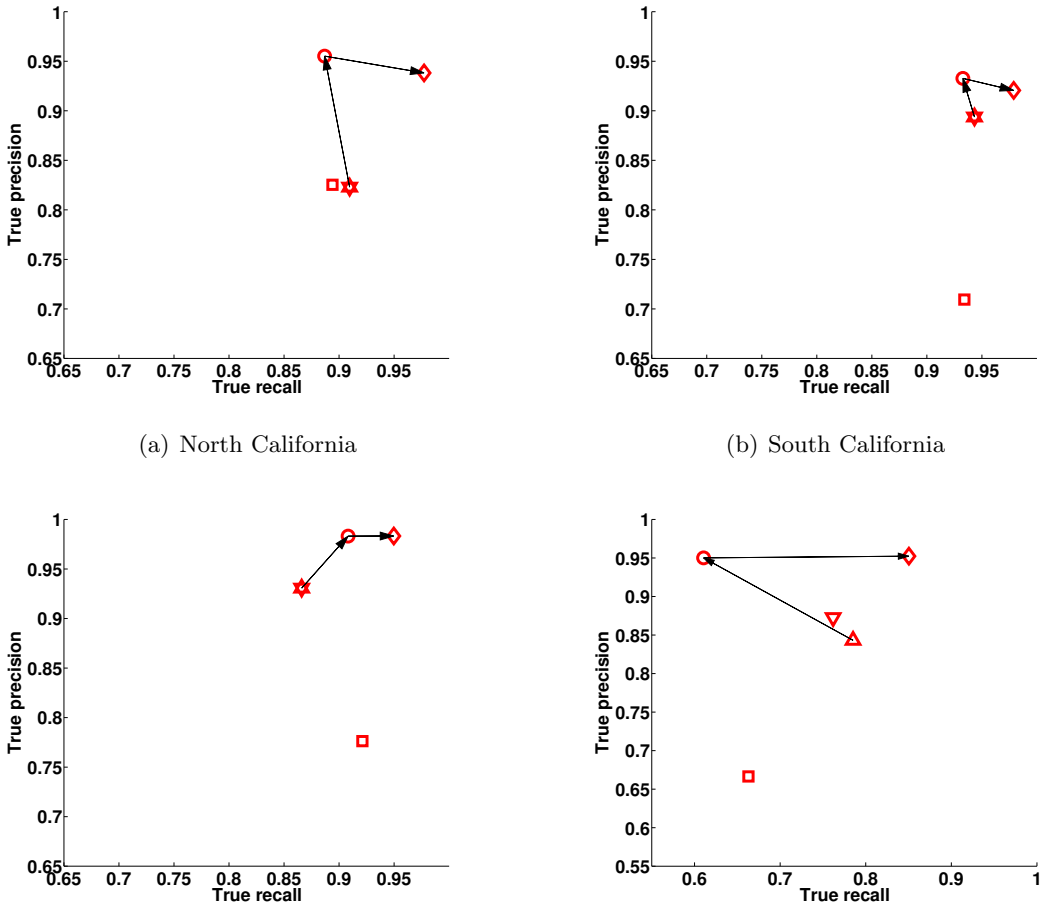
One of the key aspects of RAPT framework is selection of decision threshold to maximize G-measure (instead of the more commonly used F-measure). The choice of performance measure is critical because even though it is not possible to estimate actual precision and recall, which is necessary to optimize F-measure, the method estimates the product of actual precision and recall (upto a proportionality constant) corresponding

to each candidate threshold, which is used to optimize G-measure.

F-measure and G-measure assign equal importance to precision and recall of rare class. This conforms to joint maximization of recall and precision that is commonly used in remote sensing. However, in some applications recall may be more important (eg. diagnosis of diseases) and in other applications precision may be more important (eg. spam filtering). Weighted geometric mean can be used to assign greater importance to either precision or recall. Weighted geometric mean (weight of precision is 1 and of recall is $k$) is written as $(\text{precision} \times \text{recall}^k)^{\frac{1}{1+k}}$, where choice of $k$ is application-specific. The method presented can be easily adapted to optimize weighted geometric mean.

The RAPT framework was motivated by the applications in the land cover monitoring domain where the target class is rare, availability of gold standard labeled training samples is a challenge, and spatial neighborhoods are well defined. However, the concepts developed are more general and relevant for other application domains such as cyber-security, diagnosis of rare diseases, and fraud detection. For example, Step 1 of RAPT by itself presents an algorithm to train classification models for rare classes to optimize G-measure using imperfectly labeled training samples. Applications like spam filtering and fraud detection, where gold standard labels are hard to get but imperfect labels may be easily available, can benefit by training classifiers using Step 1 of RAPT. Moreover, if imperfect labels are available for all data instances in some applications, then Step 2 of RAPT can also be applied.

In some data sets, it may be better to directly use the results of Stage 1 (and not go all the way to Step 3). This situation can arise when (i) the classifier of Step 1 itself has a high precision and recall, and (ii) the imperfect labels have extremely poor quality (high $\alpha$ and $\beta$) and as a result applying stages 2 and 3 degrades performance. The improvement in G-measure as a result of applying stage 2 is $\frac{(1-\alpha)}{\sqrt{P(a=1|g(x)>\gamma_o^{g,y})}}$. Therefore, it is possible to estimate the improvement due to step 2 by using the estimate for $\alpha$. This estimate of the improvement factor can be used to decide if applying Stages 2 and 3 is beneficial or not for a particular application. Note that since $\hat{\alpha}$ is expected to be an overestimate, the estimated improvement is likely to be a lower bound on the actual improvement.



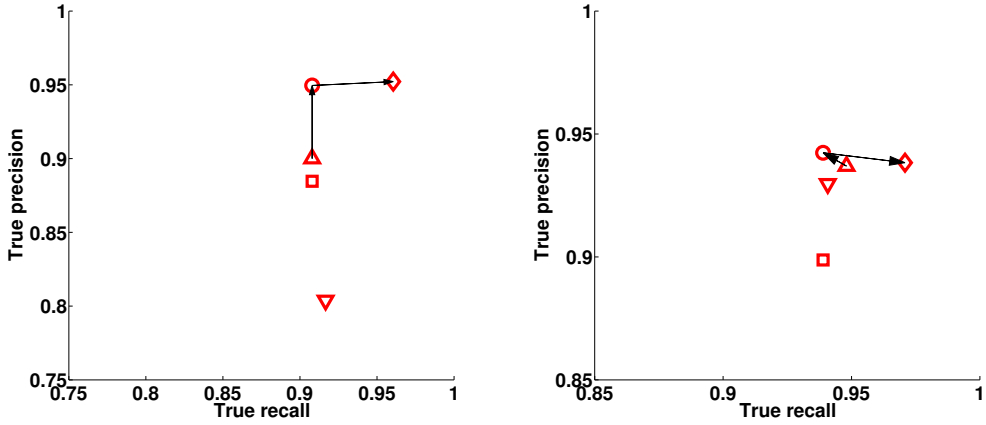

(a) North California

(b) South California

(c) Montana

(d) Georgia

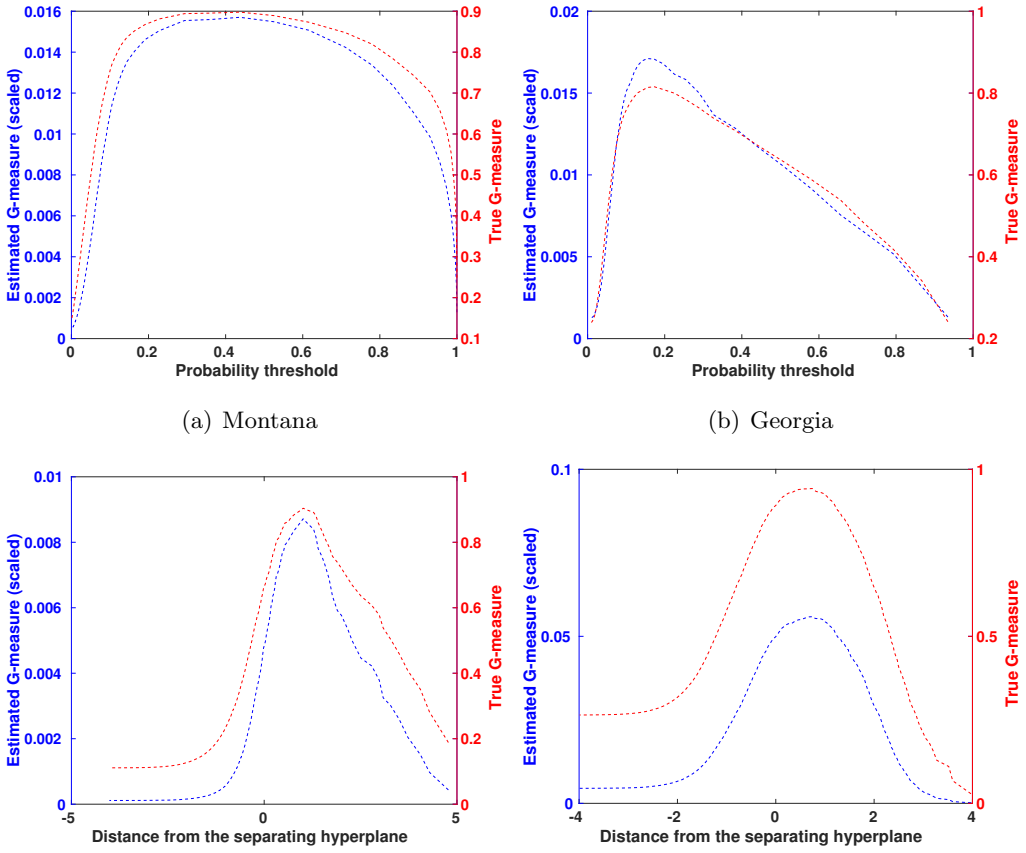
Figure 2.9: Figure shows the performance of different stages of RAPT (step 1 as *triangle*, step 2 as *circle*, and step 3 as *diamond*) for each region for burned area mapping task. The performance of classifier trained on gold standard labels is shown as *inverted triangle* and that precision and recall of the imperfect label (AF) as *square*.




(a) Rochester

(b) Mexicali

Figure 2.10: Figure shows the performance of different stages of RAPT (step 1 as *triangle*, step 2 as *circle*, and step 3 as *diamond*) for each region for urban area mapping task. The performance of classifier trained on gold standard labels is shown as *inverted triangle*. The precision and recall of the imperfect label (night-time lights) as *square*.




(a) Montana

(b) Georgia

(c) Rochester

(d) Mexicali

Figure 2.11: Comparison of true G-measure (*in red*) and their (scaled) estimates (*in blue*) using RAPT method for different values of thresholds on $g(\mathbf{x})$.

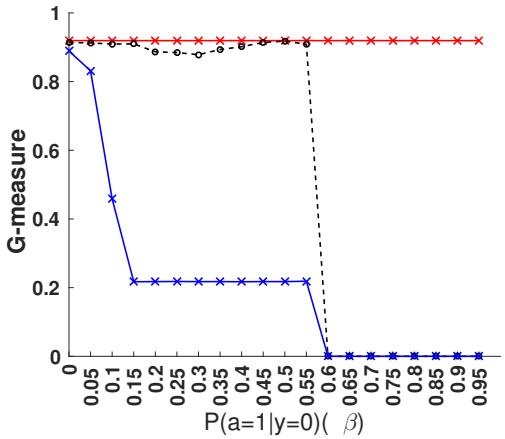



Figure 2.12: Performance of three classifiers on different training sets corresponding to different quality of imperfect labels (red: classifier built using true labels, blue: classifier built using corrupted labels but treating them as true label, black: built using the method in section 3 - RAPT stage 1). The x-axis shows the imperfection (measured as $\beta$) in the training samples for each dataset. The y-axis shows the performance (measured as G-measure) of each of the three classifiers for every dataset.

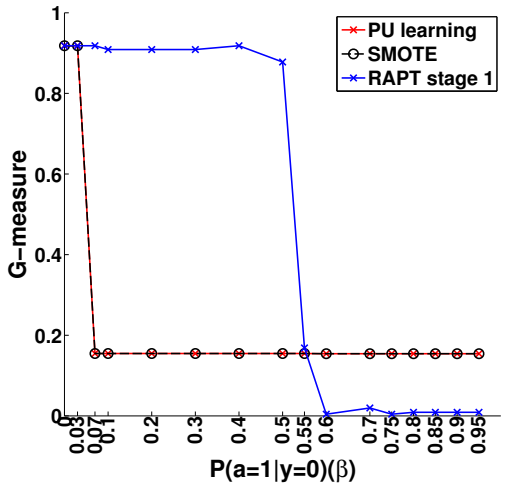
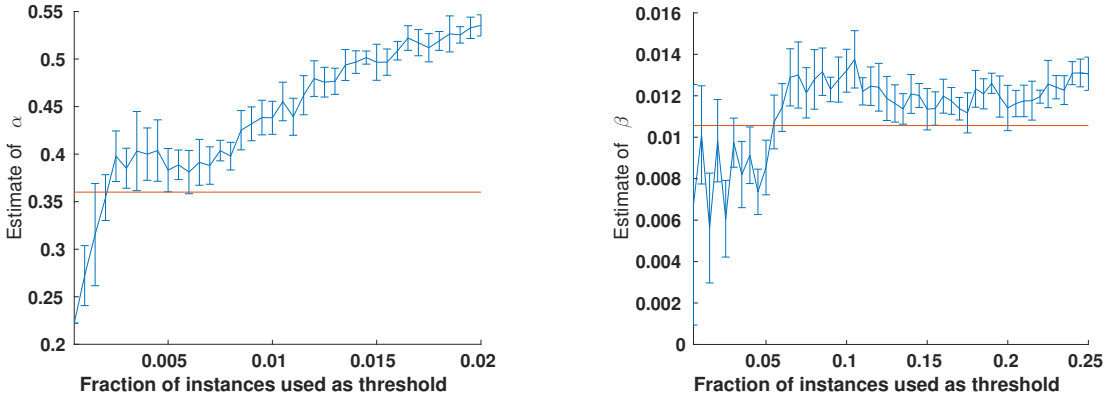



Figure 2.13: Performance of three classifiers on different training sets corresponding to different quality of imperfect labels (black: classifier built using SMOTE resampling, red: classifier built using PU learning algorithm, blue: built using the method in section 3 - RAPT stage 1). The x-axis shows the imperfection (measured as $\beta$) in the training samples for each dataset. The y-axis shows the performance (measured as G-measure) of each of the three classifiers for every dataset.




(a) Estimation of $\alpha$.

(b) Estimation of $\beta$.

Figure 2.14: The red line shows the actual value of $\alpha$ and $\beta$ for Georgia dataset. The blue curve shows the mean and standard deviation of $\hat{\alpha}$ and $\hat{\beta}$ at different fraction of instances used for estimation.

# Chapter 3

# Change Detection from Temporal Sequences of Class Labels

## 3.1 Introduction

The physical surface of the earth is undergoing constant change: new cities are being built, existing cities are expanding to accommodate population increases, forests are being cleared for agricultural use, and lakes and other water bodies are changing in their extent [31]. The growth of urban areas, in particular, has received recent attention because the resulting large-scale changes have immediate impacts on a host of environmental factors [14]. Urban areas are now expanding twice as fast as their populations [14], and the United Nations estimated that more than half of the world's population lives in cities in its *2011 Revision of World Urbanization Prospects* report [32]. Thus, the ability to monitor the extent and rate of urbanization is critical due to its far-reaching consequences for the regional environment and beyond [33]. In particular, there is a strong need for accurate, timely, and regularly updated maps of global urban extent and dynamics.

Remote sensing has enabled the acquisition of multi-spectral imagery that can be used to study the earth surface. Data sets obtained via remote sensing are at a range of spatial and temporal resolutions (there is an inherent tradeoff between the two). Mapping urban extent and growth has traditionally been performed using moderate- to high-resolution multi-spectral data (250m to 30m). In most approaches in the literature

[29], a classifier is built using manually selected training samples and is used to assign a land cover class from a finite set of classes to every pixel based on observed spectral values (the feature space). When a new image arrives for the latest time step, this classification process is repeated and every pixel is reassigned to a land cover class based on its current spectral values. Thus, every pixel is assigned a sequence of land cover class labels corresponding to the time period of image collection.

We focus on the following two questions (in the region and time interval of interest):
Q1: Which pixels belong to an urban land cover class (or any other land cover class of interest)? (i.e. what is the urban *extent*?)
Q2: Which pixels changed from vegetation to urban land cover class (or any other land cover transition)? (i.e. where is land being *converted* to urban cover?)

The ability to answer the questions above depends on the accuracy of the underlying classification map, such as GL00 [34], IMPSA [35], GRUMP [36] or MOD500 [37]. The land cover research community has investigated many sophisticated approaches, including Random Forests [38] and Support Vector Machines [39]. However, satellite data has unique characteristics that make classification difficult: classes are often mixed, the feature space is unstable (due to variability in the sun angle, atmosphere, image registration, etc.), there is multi-modality within classes (e.g. different kinds of trees), and there is a lack of high-quality training data. A recent study [29] showed that existing maps of global urban areas have disparities up to an order of magnitude, and discusses the lack of frequent map updates. Some classification maps (especially regional maps) are built with significant manual input from domain experts and are of high accuracy. However, even these detailed regional land cover class maps only have a classification accuracy between 85% to 95% [40].

To illustrate how classification accuracy will impact our ability to answer Q1 and Q2, let us consider a classifier built for a data set with two target classes and balanced training samples. Assume the classifier assigns a given test object to the incorrect class with probability $\epsilon$ and to the correct class with probability $1 - \epsilon$. For the query Q1 (identify pixels of class *1* at any given time step $t$), the accuracy of the result set will be $1 - \epsilon$. If $\epsilon$ is 0.1 (as is often the case for regional maps), the accuracy (also precision and recall) is 0.9. Therefore the result sets for Q1 are useful for the end-user looking for regions that belong to a particular land cover class.

Next, let us perform a similar analysis for query Q2 (identify pixels that changed from class *1* to class *2* between time $t_1$ and $t_2$), with the same classifier and error characteristics as the case above. Let us also assume the fraction of land surface that actually changed in this period is $p$. Due to classification error, $\epsilon$ fraction of pixels belonging to class *1* will be assigned class *2* in time $t_2$. Similarly, $\epsilon$ fraction of pixels belonging to class *2* will be assigned class *1* in time $t_1$. Thus, even when there is *no* land cover change between $t_1$ and $t_2$, $\epsilon$ fraction of class *1* pixels and $\epsilon$ fraction of class *2* pixels will be designated as changes from class *1* to class *2* and vice-versa. These incorrect labels will contribute to the false positives for a change detection query. Similarly, there will be $2\epsilon p$ false negatives. Ignoring the higher order terms in $\epsilon$, the expected recall is $\frac{p-2\epsilon p}{p}$, and the expected precision is $\frac{p-2\epsilon p}{p+2\epsilon}$. However, changes in land cover typically occur in a very small portion of a large region of study; the area changed is often less than 1% of the total area ($p \approx 0.01$). Therefore, recall $\approx 0.8$, and precision $\approx 0.05$. Thus, even for high accuracy, state-of-the-art land cover classification products, the precision of change detection maps can be as poor as 5%. The analysis above shows that when land cover change mapping is done using post-classification comparison of images, even small amounts of classification inaccuracy can significantly lower precision.

We present an innovative data mining approach to improve the class labels (associated with pixels) in land cover maps by using multi-temporal data. Our main intuition is to exploit the rich contextual information present in the temporal sequence of class labels which is not used by the classifier while generating class labels for images from different time steps. Our approach takes the original sequence of class labels as input; our task is to compute a new label sequence that is closer to the true land cover class of a pixel at the corresponding time step than the original classication sequence. Previous studies have used smoothing methods to improve classification using temporal context [41, 42] as well as spatial context [43]. In this study, we use a Hidden Markov Model (HMM) as the generative process for land cover label sequences and use the HMM for post-classification temporal smoothing to correct classification errors.

## 3.2 Related Work

We discussed use of Hidden Markov Models (HMMs) for temporal smoothing of outliers in class label sequences. HMMs have also been used as models to infer the land cover state of pixels from continuous spectral time series data as input [44, 45]. This body of literature uses HMMs to leverage temporal context for land cover classification at each time step from multivariate time series data as input. The model requires specification of a class conditional data distribution and [44] uses a multivariate Gaussian distribution for the multivariate spectral data. In our problem setting we decouple the classification and temporal smoothing tasks. We assume we are given a classifier that maps the multi-spectral data to a land cover class, and our objective is to use the temporal sequences of land cover class labels to correct any misclassification by incorporating temporal context. One advantage of decoupling these two tasks is that it allows usage of more powerful discriminative classifiers such as Decision Trees, SVM or Random forests that can learn complex decision boundaries in spectral data that generative models such as multivariate Gaussian distribution might fail to learn.

We model a latent, mixed land cover class and identify it from the temporal sequence of pure land cover class labels. Methods in remote sensing literature use spectral unmixing approaches [46] to find the fractional composition of mixed pixels in terms of pure classes. These approaches assume that the spectral data for mixed class pixels is a linear combination of spectral distributions of the pure classes. Our latent state modeling of a mixed pixel is different from these approaches as we do not assume that spectral distributions are additive in nature, but instead rely on the confusion between pure classes for a mixed pixel over multiple time steps. However, it is important to note that our approach only aims to identify mixed pixels and does not provide the exact proportion of pure land cover classes in the pixel.

## 3.3 Proposed Approach

In this section, we describe our approach for transforming an inaccurate class label sequence into a new, more accurate class label sequence. Furthermore, we describe two useful scores that can be derived from the proposed generative model for label sequences that will aid change detection queries by associating a notion of confidence with a given

pixel's sequence of labels.

### 3.3.1 Definitions

We begin by defining some terms and concepts related to the land cover mapping domain. A *pixel* is a fixed, regular-shaped spatial portion of an image. The entire image is divided into a mutually exclusive and exhaustive set of pixels. A land cover *class* or *label* comes from a finite set of classes used to categorize pixels on the earth surface (vegetation, water, etc.). A temporally ordered set of observed labels of a pixel is called a *label sequence* and represented as $c^i$. The sequence of "true" land cover state of a pixel is called its *state sequence* and represented as $z^i$.

The true land cover state may be different from the observed label due to noise in data or classifier inaccuracy. The difference between the observed label and the actual state of an object is referred to as *confusion*. The process of transformation of the material on the earth surface due to natural or human-induced actions such as wildfires and urbanization is known as land cover change. In this study, *land cover change* refers to a transition in the land cover state of a pixel occurring over a time period. Every object is assigned a land cover label based on its spectral values by the classifier. We refer to the label sequence data of all the pixels in an image as the *original classification* ($C^o$). The proposed approach assigns every object a new land cover label. We refer to this new label sequence data of all the pixels in an image as the *new classification* ($Z^1$).

### 3.3.2 Observations

The main intuition behind our methodology to modify class labels is based on the following observations in the land cover classification and change processes:
*Observation 1*: A pixel tends to remain in the same land cover states over time; therefore, change in land cover state is an infrequent phenomenon. This means that probability of transition to a different state is significantly smaller compared to self-transition.
*Observation 2*: The classification is more likely to be correct. This means that probability of misclassification is significantly smaller compared to correct classification.
*Observation 3*: The transition probability between some pairs of classes is higher than others. For example, vegetation to urban land cover change is more likely to happen

than urban to vegetation.

*Observation 4*: The confusion probability between some pairs of classes is higher than others. For example, confusion between vegetation and urban is higher than confusion between vegetation and water.

Based on the above observations, the goal of our proposed method is to generate a new classification $Z^1$ using the original classification $C^o$ as input, by modeling the observations about land cover classification and change processes. Our objective is that this method will assign correct labels to pixels that were misclassified in the original classification. In other words, $Z^1$ should have a higher classification accuracy than $C^o$.

### 3.3.3 Method

Here, we describe the proposed generative model for the observed label sequence $c^i$ of a pixel $i$. We assume that there exists a latent true land cover state $z_t^i$ for each pixel $i$ at time $t$. However, $z_t^i$ is a latent variable and we can only observe $c_t^i$, i.e., the label assigned to it by the classifier based on the observed spectral signal. If the classifier was perfect and there were no data inaccuracies/incompleteness, then $c_t^i$ will be sufficient to infer $z_t^i$. More precisely, we would set $z_t^i \equiv c_t^i$. However, classifiers make errors and satellite data has imperfections, and therefore $c_t^i$ and $z_t^i$ are sometimes different. Note that if we directly use the observed class label sequence for change detection, each confusion will be counted as a class transition, creating several spurious land cover changes.

We model the stochastic process of class confusion in the classifier with a latent confusion matrix $M$, such that $m_{kl} = P(c_t^i = l | z_t^i = k)$, i.e., the observed land cover label is $l$ when the actual land cover state is $k$. Based on Observation 2, we assume that $m_{kl}$ is highest when $k = l$. Moreover, since we will always observe some label at each object, the following constraint is always satisfied: $\sum_l m_{kl} = 1$. Observation 3 discussed different transition probabilities for every pair of classes. We model this stochastic process as a transition matrix $T$, where each entry $t_{xy} = P(z_t^i = y | z_{t-1}^i = x)$, i.e., pixel in land cover state $x$ transitions to state $y$. Based on Observation 1, $t_{xy}$ is highest for $x = y$, since change in land cover class of a pixel is a rare event. Since each pixel in state $x$ will transition to some state $y$, the following constraint is always satisfied: $\sum_y t_{xy} = 1$.

Thus, we see the above process is a doubly embedded stochastic process: an underlying stochastic process of transitions in true land cover state which is only observable through another stochastic process of class labels assigned by the classifier. Additionally, we make the following assumptions: (1) Land cover class change is a first order Markov process, i.e., $z_t^i$ does not depend on $z_{t-2}^i ... z_1^i$ given $z_{t-1}^i$, and (2) the observed classification output $c_t^i$ depends only on $z_t^i$. Motivated by the observations and assumptions above, we represent the generative model for land cover class sequences using a first-order Hidden Markov Model (HMM) with $k$ states corresponding to the land cover states, $s$ symbols corresponding to the class labels, a transition matrix $T$, and an emission matrix that corresponds to the latent confusion matrix $M$. Our next task is to infer the latent variables $M$, $T$, and $Z^1$ given the sequences in $C^o$, which we perform in two phases: parameter learning and inference.

**Parameter Learning**

If the number of classes $k$ is small, a domain expert may be able to provide good estimates for $T$ and $M$. One can also provide an uninformative prior estimate for these matrices, with the final estimates being computed using an expectation-maximization algorithm that maximizes the posterior probability of state sequences [47]. Due to the issue of missing labels (e.g. because of missing spectral data), some of the $c_t^i$ are assigned a missing class label. Therefore, we allow an additional observed label corresponding to missing data. In principle, a missing label has no information about the true state and observing a missing label is equally likely for every latent land cover state.

**Inference Step**

We now discuss how to find the new class labels that are "most" likely given the observed label sequence $c_i$ and the model parameters $M$ and $T$. Note that there is no "true" sequence to be found and for a practical solution we look for a sequence that maximizes some objective function. Here, we consider the most likely state sequence as the one that maximizes the posterior probability $P(Z^1|C^o, M, T)$. The solution of this formulation is discussed in detail in [47]. Other optimality criteria such as "most" probable individual states are less suited in our problem as they may assign states that form infeasible sequences due to the presence of unlikely transitions. Next, we use our probabilistic

model for class label sequences to derive two useful scores that will aid change detection queries.

### Confidence Score

Our framework makes certain assumptions on the generative process of the land cover label sequences. If the observed sequences for pixels violate these assumptions, the possibility of classification errors in $z^i$ is higher. Motivated by these issues, we define confidence score, a measure that indicates how well the HMM model fits the pair of an observed and hidden sequences. The score can also be used to partition the pixels into two subsets of high and low confidence scores. We are more confident that $z^i_t$ represents the true state for the pixels with higher confidence compared to pixels with low confidence subset.

We propose to use the logarithm of the joint probability of observed and hidden sequence, i.e., $P(z^i, c^i|M, T)$ as a measure of our confidence on the reconstructed state sequence. *A high joint probability implies higher likelihood of $z^i$ and $c^i$ under our assumptions on $M$ and $T$.* Therefore, pixels which have either many missing or undefined labels, or high confusion between two or more classes or a combination of these characteristics will be assigned low joint probability values. Under the Markovian assumptions in our model, it can be computed as

$$\text{Conf}(i) = \sum_{t=1}^{n} \log(m_{z^i_t, c^i_t}) + \sum_{t=0}^{n-1} \log(t_{z^i_t, z^i_{t+1}})$$

### Change Score

For any given classification product ($C^o$ or $Z^1$), query Q2 finds the pixels for which the label is $c1$ at time step $t_1$ and the label is $c2$ at time step $t_2$. Often the cardinality of the query result set is large, while the end-user is only interested in seeing a few samples of the result. In such cases, it is more useful to provide a subset of results with a higher precision than a random subset from the result set.

We propose to use $P(z^i_{t1} = c1, z^i_{t2} = c2|c^i, M, T)$ as the *change score* associated with every pixel of the query result. If $t_2 = t_1 + 1$, then this probability is equivalent to $\xi_{t1}(c1, c2)$, which is used in the Baum-Welch algorithm [47], and a dynamic programming-based solution is known for computing it. However, for our purpose, we

need to generalize $\xi_{t1}(c1, c2)$ to $\xi_{t1,t2}(c1, c2)$ which corresponds to $P(z^i_{t1} = c1, z^i_{t2} = c2|c^i, M, T)$. $\xi_{t1,t2}(c1, c2)$ can be computed as $\frac{P(c^i, z^i_{t1}=c1, z^i_{t2}=c2)}{P(c^i)}$.

## 3.4 Data and Materials

Our proposed method takes as input a sequence of classified land cover maps. In this section, we describe the data and methods that we use in this study to generate the sequence of classified maps (i.e. the input to our method). We begin by describing the multi-spectral satellite imagery that serves as input for the classifier, and then describe the classifier and how it is trained.

### 3.4.1 Landsat Data

We used satellite imagery from the Enhanced Thematic Mapper (ETM+) sensor on board the Landsat-7 satellite. Landsat-7 is the latest in a series of Landsat satellites, which have maintained a continuous record of the Earth's surface since 1972. ETM+ is a multi-spectral radiometric sensor that records eight spectral bands of data with varying spatial resolutions (30m spatial resolution for red, green, blue, near infrared, and two bands of medium infrared; 60m for thermal infrared; and a 15m panchromatic band). Landsat-7 data is available for public download from the U.S. Geological Survey [48]. For this study, we selected the city of Belo Horizonte, located in the state of Minas Gerais in southern Brazil. Belo Horizonte is the third largest city in Brazil and one of its fastest growing cities [49], thus providing a rich dataset for evaluation of the proposed algorithm. Images were collected bi-annually, corresponding to the dry season in that region which occurs in March-April and July-August, from the years 2003 through 2012. The extent of the region is $19.5°S$ to $20°S$ and $44.16°W$ to $43.75°W$, and it contains 2,868,495 pixels at 30m spatial resolution.

### 3.4.2 Base Classifier

We implemented a classification module based on the methodology proposed by [40]. This approach, recently developed at DLR (the German Aerospace Center), is a state-of-the-art method for urban mapping. The approach overcomes many of the challenges in classifying remote sensing imagery (e.g. multi-modality, lack of training data) using

sophisticated innovations. The classifier in [40] is an ensemble of binary decision trees; each tree is trained for a different class using the labeled training samples. The binary trees are then ordered by land cover class using domain expertise. Finally, each Landsat image is independently classified into three categories: water (W), vegetation (V) and urban (U). If the spectral data for a pixel is missing in an image or the pixel is not assigned a class by the ordered binary decision trees, then a missing label is assigned to that pixel for that image. [40] were able to develop urban maps for 27 mega-cities globally with little training effort. (Though the authors also considered change maps in their paper, their focus was on general trends across decadal time scales, not pixel-level accuracy.)

### 3.4.3 Validation Imagery

Land cover mapping research is often impeded by the lack of gold standard ground truth data, in our case pixel-level classification labels. To overcome this issue, in this study we take advantage of the high-resolution imagery available in Google Earth to interpret classified land cover labels, as other urban mapping studies have done [29]. The high-resolution imagery, which is orders of magnitude more detailed than Landsat, is tagged with the time of observation. For most regions of the globe, there is usually only a few (if any) high-quality cloud-free images available. To validate a given pixel's class label, we carefully examine the high-resolution imagery for agreement with the label; it is important to note that an additional criterion for agreement is that the time of observation of the validation imagery and classified image are similar. Henceforth, when we use the term *validation imagery*, we are referring to high-resolution imagery from Google Earth. Note that while imagery can be visualized in Google Earth, the underlying multi-spectral data is from commercial satellites and generally not freely available for analysis.

| Classification data | Set of all pixels | Set of pixels with high confidence score |
|---|---|---|
| $C^o$ | 9.5% | 7.12% |
| $Z^1$ | 1.89% | 0.73% |
| $Z^2$ | 1.65% | 0.95% |

Table 3.1: Fraction of pixels with different land cover class in August and September, 2008.

## 3.5 Evaluation and Discussion

### 3.5.1 Classification Accuracy

In this experiment our goal is to show that (1) a sequence of land cover maps, while individually accurate ($\geq 90\%$), has errors which may lead to identification of spurious land cover class changes, (2) the temporal context is leveraged by our method to improve the class labels and thus avoid these spurious changes and (3) we are also able to assign a confidence score to every pixel and the classification accuracy of both $C^o$ and $Z^1$ is higher for the subset of pixels with a high confidence score than the subset with low confidence score.

The data consists of two multi-spectral Landsat images one month apart (August and September 2008) for the city of Belo Horizonte. Each pixel of the two images is assigned a label from the target classes (W, V or U) based on their spectral attributes.

First, we consider the scheme of bi-temporal post-classification comparison to identify pixels that have different class labels in $C^o$. If these pixels (that were assigned different labels in August and September 2008) are treated as land cover changes, then Table 3.1 shows that 9.5% of the pixels changed their class in a period of one month. However, land cover change is a relatively rare event (across large spatial areas) and one expects the annual rate of land cover change to be under 1%. Thus, it is reasonable to consider these 9.5% pixels with land cover change in the short duration of a single month to be spurious due to classification errors.

In the absence of ground truth, for this experiment we rely on the percentage of changes in one-month period as a reasonable metric to compare the accuracy of any two

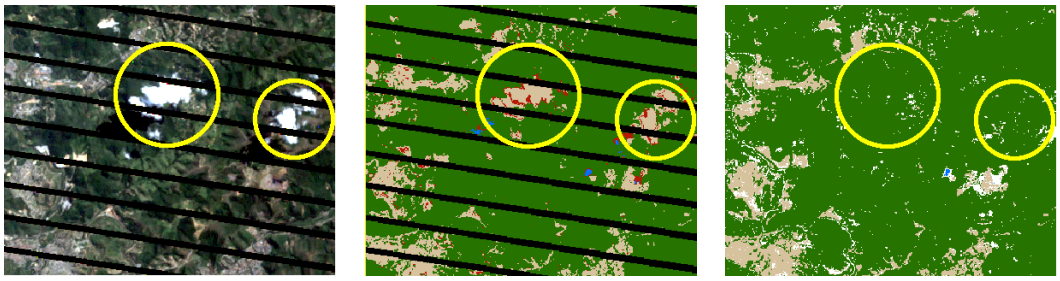
land cover maps. Ideally, one expects this to be 0 (or close to 0) for a perfect classifier. For an imperfect classifier, we expect this percentage to *decrease* as the classification accuracy *increases*. (We previously discussed the connection between classification accuracy and spurious changes in Section 3.1.)

We used the proposed approach to obtain a new label sequence given a label sequence from classifier. Thus, the class label for a pixel for August, 2008 may be different from its label in the original classification. Next, we repeat this step for the same temporal sequence data with the class labels for August 2008 swapped with the class label from September 2008. September 2008 gets a new label which may also be different from its original label. If temporal context is playing a positive role in improving the class labels, then we expect that in case one of the labels (for August or September) was incorrect and is getting correctly reclassified in $Z^1$ the number of mismatches will reduce. Table 3.1 shows that the number of transitions in one month period is reduced from 9.5% to 1.89%. We observed similar results when we repeated this experiment using data from different years.

Next, we partitioned the set of all pixels into high and low confidence score subsets and for 80% of the pixels which are in the high confidence subset our model improved classification accuracy and reduced the spurious transitions from 7.12% to 0.73%.

### 3.5.2 Correcting & Imputing Labels Due to Poor Data

Remote sensing data is often plagued with noise (due to atmospheric interference such as clouds and aerosols) and missing data due to instrument malfunction. Therefore, a given classified image $C^o$ often has labels that are both inaccurate (due to noise) and incomplete (due to missing data). To illustrate both these issues, Figure 3.1(a) shows a Landsat image in which we can see the presence of clouds (highlighted with the yellow circle) and missing data (shown as black stripes) on the earth surface. Figure 3.1(b) shows the classified map ($C^o$) corresponding to Figure 3.1(a), where clouds have been misclassified as urban and Figure 3.1(c) shows the classified map ($Z^1$), where our method has reassigned the cloudy region to the V class and imputed missing labels using its temporal context.



(a) Landsat image in Belo Horizonte.

(b) Classified map.

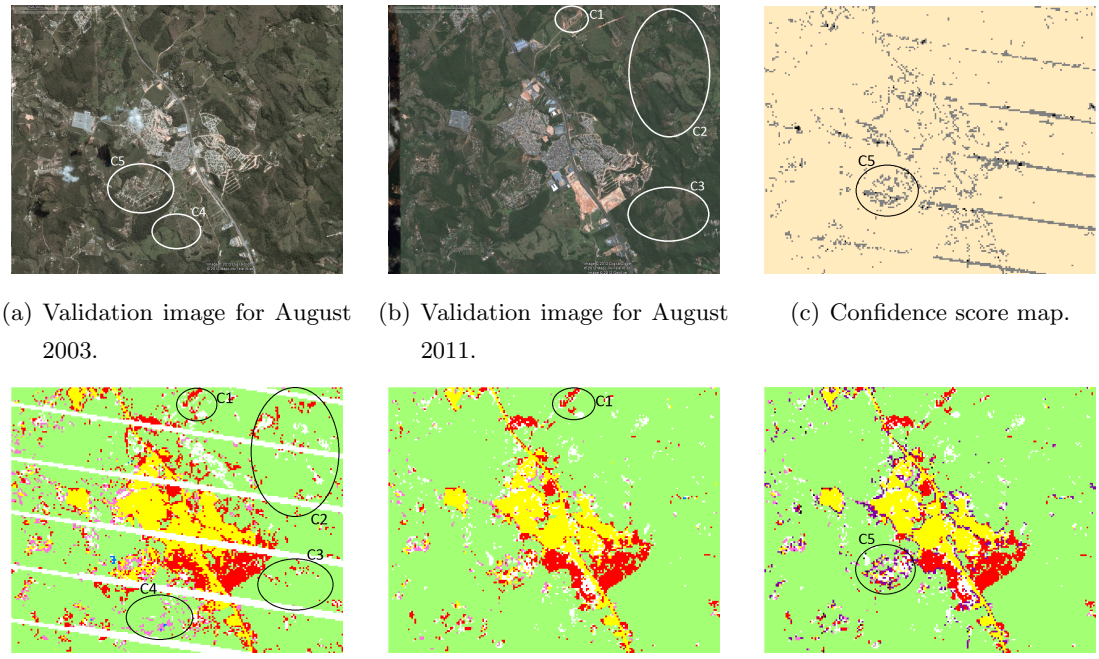
(c) Classified map $Z^1$.

Figure 3.1: These figures show the issues of noise and missing data, and how our proposed method is able to correct the labels caused by these issues.

### 3.5.3 Change Detection

Given two classification products, $C^o$ and $Z^1$, one can create a change map for any pair of time steps $(t_1, t_2)$. In this section, we will show that the change map prepared using $Z^1$ is better than one prepared from $C^o$. In particular, there is an improvement in both the number of false negatives and false positives.

Figures 3.2(a) and 3.2(b) show the validation imagery of a region in the city of Belo Horizonte that has instances of urbanization between the years 2004 and 2011. Figure 3.2(a) corresponds to the image from April 20, 2003 and Figure 3.2(b) corresponds to the image from August 22, 2011 for the same region. Figure 3.2(d) shows the change map produced using the bi-temporal post-classification comparison method for the images from years 2004 and 2011. This map has nine unique categories of transitions corresponding to the 3 classes (V: vegetation, U: urban and W: water), but has only four dominant categories: V $\rightarrow$ V (in green), U $\rightarrow$ U (in yellow), V $\rightarrow$ U (in red) and U $\rightarrow$ V (in pink). Here, we make some observations upon studying Figure 3.2(d) in the context of Figures 3.2(a) and 3.2(b). The classifier is usually accurate and we can see that the vegetated areas and urban areas (as they appear from the validation imagery) are typically assigned the V $\rightarrow$ V or U $\rightarrow$ U labels. We also see that this region is dominated by unchanged (V $\rightarrow$ V and U $\rightarrow$ U) pixels. The missing data issue is seen as white stripes on the change map because change cannot be determined for a pixel if either of the two images have a missing label. Among the different change categories,




(a) Validation image for August 2003.

(b) Validation image for August 2011.

(c) Confidence score map.

(d) Change map using original classification $C^o$.

(e) Change map using multi-temporal context-based classification $Z^1$.

(f) Change map using multi-temporal context-based classification $Z^2$.

Figure 3.2: These figures show the change detection for a region of study between 2003 and 2011.

V $\rightarrow$ U and U $\rightarrow$ V are the dominant change types, and between them, V $\rightarrow$ U change is significantly higher in number than U $\rightarrow$ V. This is expected as typically vegetated areas are converted to urban land and not vice-versa.

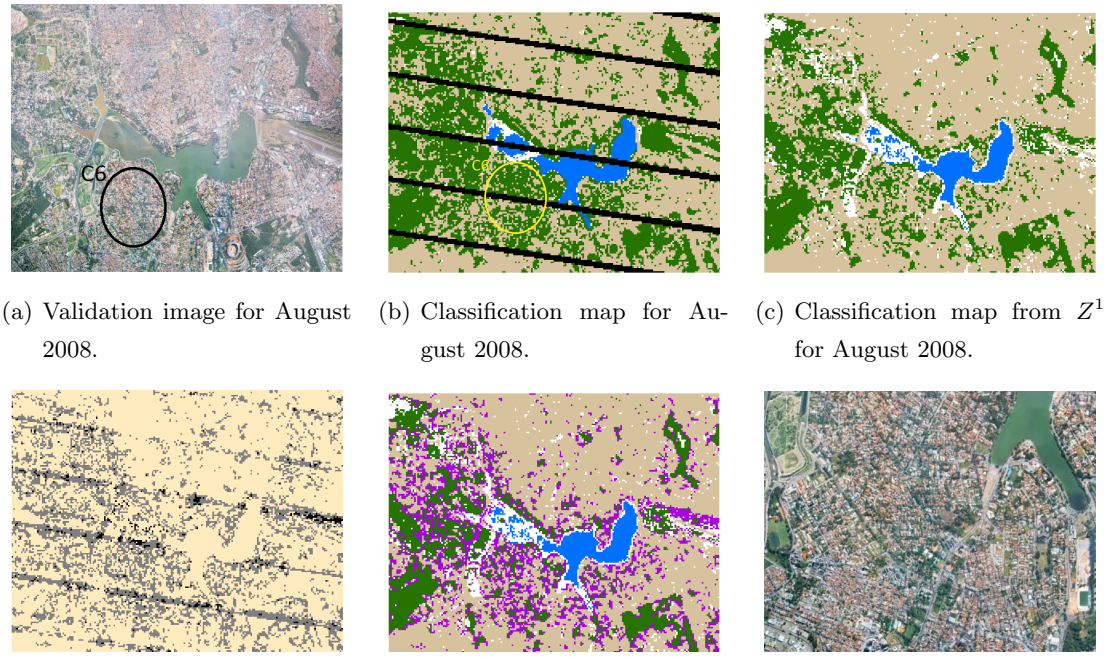
This region has two large clusters (of several hundred pixels) that are marked as conversion from V $\rightarrow$ U. The validation imagery for the corresponding region also shows that the land surface was vegetated in 2003 and was barren by 2011. In addition to these large changes, we see some moderate sized clusters consisting of 20-100 pixels (such as the example in C1) and of salt-and-pepper distribution of changes (such as the examples in C2 and C3). The other dominant category of change is U $\rightarrow$ V (marked as pink pixels). There are a few small clusters of 5-10 pixels that are marked as U $\rightarrow$ V (such as the examples in C4). Finally, we see that several pixels on the boundary of two land cover types have been marked as changed, either from U $\rightarrow$ V or vice-versa.

Next, we focus on Figure 3.2(e) which shows the change map produced by comparing the class labels of $Z^1$. Our first observation is that the number of changes (both V $\rightarrow$ U and U $\rightarrow$ V) is smaller for the new change map compared to Figure 3.2(d). The V $\rightarrow$ U changes reduce from 2880 to 1935 and the U $\rightarrow$ V reduce from 846 to 360. The relative reduction in U $\rightarrow$ V is higher because most of the changes in that category were spurious and these spurious changes are expected to decrease with a more accurate classification. To verify that the reduction in number of changes actually corresponds to a reduction of false positives and not an increase in false negatives, we examined validation imagery for the areas in circles C2 and C3 that are marked as changed from V $\rightarrow$ U in $C^0$ and are not changed in $Z^1$. In our analysis, we found that the validation imagery corroborates better with the new map from $Z^1$ and the pixels that were previously labeled as change (in $C^0$) were spurious transitions in most cases. Further evidence supporting this claim is the fact that most of the changes in large or moderate sized clusters such as C1 persist in the new change map, while most of the changes inside C2 and C3 are removed. C1 has a change from V $\rightarrow$ U which is also visible in the validation imagery, while C2 and C3 have a salt-and-pepper distribution of V $\rightarrow$ U changes with no clear evidence from the validation imagery.

### 3.5.4 Mixed Pixel Modeling

Remote sensing data is recorded for fixed-size spatial units (pixels) and therefore pixels that contain a mixture of land cover types are inherently present in these data sets. This occurs even with Landsat, which is one of the highest resolution publicly available data sources (Landsat's 30m pixel size roughly translates to an area of 0.22 acres on the ground). Naturally, the occurrence of mixed pixels increases significantly for coarser scale data sets such as MODIS (250m) and SPOT (1km). However, most land cover classifiers are trained with pure classes because mixed pixels have tremendous heterogeneity and it is infeasible to generate a representative set of training samples that adequately captures all kinds of mixed pixels. Therefore, the ability to tag mixed pixels (whether they appear on boundaries or in clusters) as belonging to a mixed *class* using output from classifiers that were trained for only pure classes represents a significant advance in land cover mapping.

Our model for land cover class label sequences assumes that the pixels are either W,




(a) Validation image for August 2008.

(b) Classification map for August 2008.

(c) Classification map from $Z^1$ for August 2008.

(d) Confidence score map, black implies low score.

(e) Classification map $Z^2$ after including mixed class in model for August 2008.

(f) Zoomed-in image for the area in C6 in August 2008.

Figure 3.3: These figures show the classification maps for a region of study in Belo Horizonte between 2003 and 2011.

V or U. For example, Figure 3.3(a) shows the validation imagery for a spatial region around an urban area. Running the classifier trained on the set of target classes (W, V and U), a classification map for this region is shown in Figure 3.3(b). The classifier is typically able to identify the water, urban and vegetation classes, but we see that it has misclassified several pixels as vegetation in an urban area inside circle C6 (as it appears from validation imagery). As was previously discussed in Section 3.3, confusion between classes is one of the reasons for this; Figure 3.3(c) which shows the new classification map has significantly reduced the number of pixels classified as vegetation in that region. However, we see some of the pixels inside C6 are still misclassified.

To investigate this issue, we computed the confidence score for each pixel of this region. Recalling the discussion from Section 3.3.3, this score measures the conformity of a given sequence to the model parameters and assumptions. Figure 3.3(d) shows a

map of confidence scores for the pixels in the region. This map clearly shows that C6 and some other subregions in the region of study have very low probability compared to the rest of the image. We examined these low probability regions and found that these pixels have a significant (and almost equal) proportion of both vegetation and urban labels in their sequences. The validation imagery for these regions also reveals that these pixels are neither completely vegetation nor urban, but rather a mixture of the two classes. If our model attempts to assign these pixels to either the V or U class, the confidence score would be very low, as they have much higher confusion than expected by the HMM model.

Next, we extend our approach to include the concept of a mixed class (M), which is a latent class in our model that is equally biased towards observing the V or U class labels in its sequence. Figure 3.3(e) shows the classification output ($Z^2$) after including M. We see that several pixels in the low probability areas are now assigned to M. This novel, mixed class replaces most of the inaccurate vegetation class labels, while the actual vegetated areas are classified as vegetation. Moreover, we see that pixels at the boundaries of two land cover classes are also assigned to M. This is not surprising as boundary pixels are typically expected to have a proportion of both classes present in them and also confirms that the latent mixed class in our approach actually captures the pixels that physically consist of a mixture of two classes on the ground. The above experiment demonstrates the capability of the extended model to identify novel, latent classes from sequences of pure class labels by using the temporal context.

We also found that many spurious changes in classification maps tend to occur with mixed pixels, and modeling a latent mixed class avoids identification of these spurious changes to a large extent. We return to the change detection experiment (Section 3.5.3) to illustrate that mixed class modeling can actually reduce some of the spurious changes occurring at the boundaries of land cover classes and in regions of mixed pixels. Figure 3.2(f) shows the change map produced after re-classification with mixed class modeling $Z^2$. The M $\rightarrow$ M transition is prominently visible (in deep purple color) along the boundaries of the V-U regions. Figure 3.2(c) shows the confidence score for the pixels and we can see that the boundaries between land cover types are assigned a low confidence score. Moreover, the region inside C6 has a dense concentration of low probability pixels. Validation imagery in Figure 3.2(b) confirms that these pixels are

similar to the mixed pixels seen in Figure 3.3(a). The new change map from $Z^2$ correctly assigns these pixels to a M $\rightarrow$ M transition, most of which were earlier inaccurately identified as changes in Figure 3.2(e).

## 3.6 Concluding Remarks

We applied data mining methods to advance the state-of-art in land cover change mapping by improving the existing classification products. In particular, we proposed an HMM-based generative model for land cover class label sequences and used it to infer new, more accurate land cover state sequences. Case studies on real data demonstrate that the proposed generative model is able to leverage the temporal context of class labels to improve classification. Furthermore, we used the probabilistic model to compute useful statistics, namely the confidence and change scores, which can be leveraged while analyzing change detection queries.

The goal of a semi-supervised global-scale land cover change detection system involves many challenges and this work is a step towards its realization. We explored the use of temporal context for improving land cover classification using an HMM-based model. HMM restricts the duration in same state to a geometric distribution. Future work will explore use of models such as Hidden Semi-Markov model that allow the explicit modeling of duration in the state before next transition. Another direction for research is to develop models that also use the spatial context of labels to correct classification errors. Moreover, the proposed approach was used for analyzing small regions of the size of a single city. Learning of transition and confusion probabilities is impacted by the size of selected region and future work will investigate the sensitivity to this parameter. Finally, for some regions multiple classification products are available, each with its own strengths and weaknesses. The model can be extended to integrate these multiple sources of class information in a principled framework to achieve better land cover classification than what can be achieved using a single source.

# Chapter 4

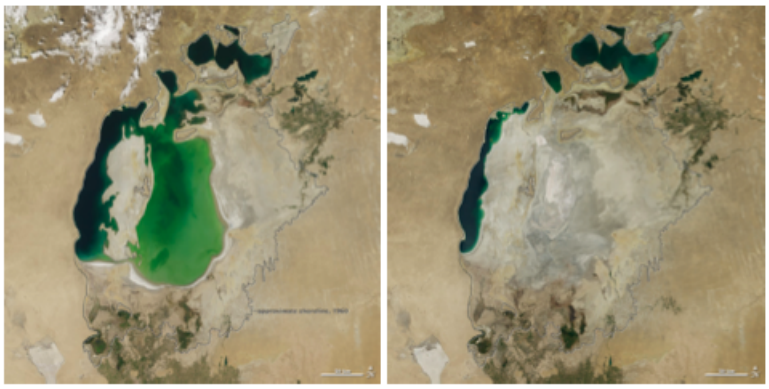
# SELPh: Simultaneous Estimation of Labels and Physical Properties

## 4.1 Introduction

Freshwater from lakes plays an essential role in supporting a variety of human needs, such as drinking, agriculture, and industrial development [50]. Lakes are dynamic in nature, they shrink, expand, or change their appearance, owing to a number of natural and human-induced factors. As an example, the Aral Sea has been steadily shrinking since the 1960s due to the undertaking of irrigation projects (see Figure 4.1), which has resulted in the collapse of fisheries and other communities that were once supported by the lake, and has further altered the local climatic conditions [51]. Global mapping and monitoring of the extent and growth of surface water bodies such as lakes is thus important for assessing the impact of human actions on water resources, as well as for conducting research that studies the interplay between water dynamics and global climate change [52–54].

There are primarily two approaches for lake surface monitoring. The first one is based on aerial and field surveys, which is extremely labor intensive and therefore infeasible for regularly updated global-scale monitoring. The other approach uses machine learning techniques for mapping the spatial extent of lakes using multispectral reflectance data from earth observing satellites [53, 55–57].

However, classifying pixels into water and land categories using classification models

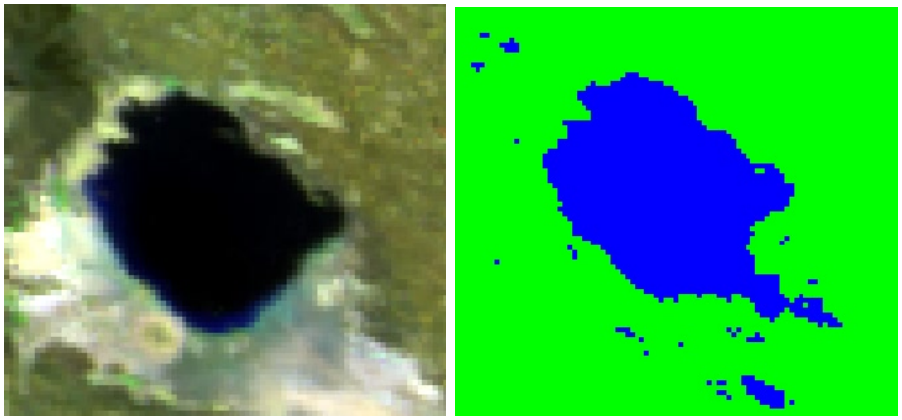


(a) August 25, 2000 (b) August 19, 2014

Figure 4.1: Change in Aral Sea surface between 2000 and 2014.

faces several challenges. In the multispectral feature space, water and land bodies can look very different at different locations (due to spatial heterogeneity), across time steps (due to seasonal changes) and even on consecutive dates (due to variations in atmospheric conditions such as clouds and aerosols). Hence, even the best classifiers (trained using high-quality, hand picked training samples) can misclassify some land pixels as water and some water pixels as land [57]. An illustration of the impact of the such class confusion is presented in Figure 4.2 that shows (a) an example of a false color composite (reference image) and (b) a classified image of Lake Abbe, Ethiopia for the same time step. The figure shows that some patches of land (as indicated by absence of water in reference image) have been incorrectly classified to water class by the classifier used to create the classified map.

One possible approach to address these issues is to enhance imperfect classification maps of multi-temporal gridded data by using some implicit information related to the phenomena under consideration. A well known example of the above approach is the spatial window majority filtering [58] that is frequently used for image de-noising. It leverages the fact that adjacent pixels in the image are more likely to belong to the same class (this is also known as the first law of geography). In this method the majority class of a sliding spatial window is assigned to the center pixel. Similarly, Markov Random



(a) False color composite (b) Classified map with a large number of classification errors

Figure 4.2: Misclassifications in a lake map due to confusion between target classes in feature space. The pixels classified as water are shown in *blue*, and as land are shown in *green*.

Field based approaches have also been used to produce homogeneous classification output that prefers same label for neighboring pixels and penalizes neighbors with different labels. *While these approaches [58–60] are effective in removing salt-and-pepper noise, they fail when there exists a significant level of spatial and temporal auto-correlation in the noise and missing data itself.* This happens frequently in remote sensing images due to seasonal variations (that can result in temporally auto-correlated noise) and atmospheric conditions such as clouds and aerosols (that can result in spatially auto-correlated noise) [61]. For example in Figure 4.2 one can see spatially correlated noise due to clouds that result in coherent patches of land pixels being incorrectly classified as water *blue*.

We focus on applications where the classification output is constrained by some physical properties of the phenomena under consideration. We present a general approach that can leverage such constraints to address the limitations of traditional classification enhancement techniques mentioned above. As an example, in the application of lake surface monitoring, one such property is that locations at a higher depth in the lake have to be filled with water at a given time if any location at a lower depth has been filled with water at that time. Figure 4.3 shows an illustrative example with 4 locations

$(A,B,C,D)$ of a lake at different depth. The physical shape of lake surface enforces that if location $C$ is water then location $D$ has to be water.

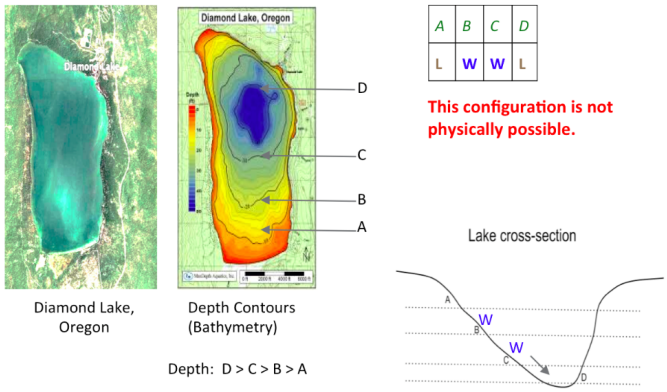



Figure 4.3: Illustration of constraints on classification output due to lake physics. The location $D$ should be labeled water before locations $B$ and $C$ can be labeled as water due to lake geometry constraints.

If the elevation information is available (eg. in the form of depth contours), then it is possible to correct the imperfections in the labels simply by changing all labels above a certain height to land and below it to water such that it minimizes the number of disagreements with the input classification. (Figure 4.4 presents a simple approach to select the optimal height for each time step.)

However, in practice the precise information about the depth of locations in unavailable at appropriate spatial resolutions and at a global scale. The framework presented allows us to make use of elevation-based constraints even when there is complete absence of elevation information.

## 4.2 Problem Setting

*Objective:* Leverage the physical properties of lakes to improve the dynamic maps of their spatial extent.

*Input:* Raster thematic maps $(P)$ of spatial extent of lakes, predicted by some "imperfect" classification model, over multiple time steps spanning several years. Each pixel

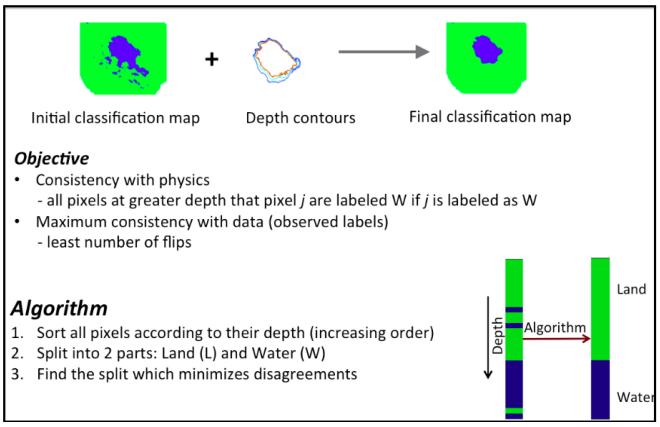



Figure 4.4: An approach to improve classification accuracy by constraining the classification output based on depth ordering.

has been assigned to one of the three classes: water (W), land (N) or missing (M).

*Output:* More accurate raster maps for each time step, produced by correcting misclassified instances and imputing missing labels. The label assignment in the output maps should be consistent with elevation-based constraints.

*Constraints:* Lake geometry constrains that if a location in the lake surface is assigned to water class (W) at time $t$, then all locations (connected to it) at greater depth should also be assigned to W for that time $t$.

In this study our goal is to develop the classification enhancement method and we do not discuss the algorithms for learning the classification model that provides the "imperfect" input classification. In principle the proposed algorithm can work with any input classification coming from a model that provides reasonably high classification accuracy.

Due to misclassifications, the initial maps typically show inconsistency with law of physics, i.e. there exist pairs of locations ($loc_1$,$loc_2$) for which at time $t_1$ $\{p^{t_1}_{loc_1} = \text{W}$ & $p^{t_1}_{loc_2} = \text{N}\}$ implying that $loc_1$ has greater depth than $loc_2$. This is contradicted at time step $t_2$ where $\{p^{t_2}_{loc_1} = \text{N}$ & $p^{t_2}_{loc_2} = \text{W}\}$ which implies that $loc_2$ has greater depth than $loc_1$. The final output should resolve all such contradictions and produce a physically consistent labeling.

## 4.3 The SELPh Approach

First, we present an algorithm for classification enhancement in scenarios where input classification maps have perfect classification accuracy but may suffer from large amounts of missing labels. This approach uses a graph formulation of the input classification maps to infer relative depth ordering and later uses the inferred depth ordering for imputing missing labels. However, in practice the input classification maps are plagued with misclassifications as a consequence of confusion between classes. This approach to infer depth order does not work for such scenarios. Next, we present the SELPh algorithm that allows simultaneous estimation of depth ordering and classification enhancement from classification maps with misclassified instances.

### 4.3.1 Correct but incomplete multi-temporal image classification

*Correct but incomplete* image classification refers to an input classification in which the class label $p_i^t$ for pixel $i$ at time step $t$, when available, is always correct. However, for several pairs of ($i$,$t$) the label is missing in the input classification. The goal here is to correctly impute the missing labels of these instances.

In our approach, we first estimate the depth order among pixels from the incomplete input classification. Once the most likely depth order is obtained, then for each time step the missing labels are imputed based on the estimated depth order and the input labels of the labeled instances.

**Estimating depth order** *Given a correct but incomplete image classification at every time step, the input labeling bears information on the relative depth order between pixels.* For example, if pixel $i$ is labeled as W and pixel $j$ is labeled as N at any particular time step $t$, then it is confirmed that pixel $i$ is at a greater depth than pixel $j$. To obtain the depth ordering, we first construct a directed graph $G = (V, E)$, where the set of vertices ($V$) corresponds to the pixels of the image and the set of edges ($E$) capture the relative depth relationship between a pair of pixels; the edge $e_{ij}$ from node $i$ to node $j$ exists *iff* at some time step pixels $i$ is labeled as W and pixel $j$ is labeled as N. Since the input labeling is "correct", it is expected to follow the law of gravity at all time steps, i.e. there would be no contradictions regarding the ordering between two pixels across

different time steps. In graph $G$, this would imply that for any pair of nodes $(i, j)$ only one of the two directed edges can exist: $e_{ij}$ or $e_{ji}$. In fact, graph $G$ is a directed acyclic graph for this problem setting.

We formulate the problem of inferring the depth ordering as one of arranging the vertices $V$ such that all the edges in $G$ are forward edges in the ordering, i.e. for all edges $e_{ij} \in E$ agree with the depth order. The depth ordering among pixels is estimated using topological sort on the graph $G$. The topological sort problem is defined as: given a directed acyclic graph $G = (V, E)$, find a linear ordering of vertices $(V)$ such that for all edges $e_{ij} \in E$, $i$ preceeds $j$ in the ordering. We refer the reader to [62] for the details of the topological sort algorithm [1]

In reality multiple pixels may have the same depth, i.e. belong to the same depth contour. Due to such grouping structure present in data, the ordering relationship among pixels is more appropriately represented by a bucket order rather than a total order. A bucket order is a special case of partial orders in which each bucket consists of multiple entities (eg. in our case pixels) with no order among themselves and there exists a total order among the buckets. In fact, topological sort on $G$ in our application gives a bucket order, in which each bucket of the bucket order corresponds to a depth contour of the lake.

**Estimating missing labels** Since all the members of a bucket are at the same depth, in the output labeling for any given time step they will either be all W or N. Moreover, since the input classification is always correct (with missing labels), for any given time step, all the *initially labeled* pixels of a bucket would have the same label (either W or N). Thus, the label of buckets with any labeled member pixel can be inferred, and subsequently the pixels with missing label in these buckets are assigned the label of their bucket. However, it is possible that no member of a bucket is labeled in the input classification at a particular time step. The labels of such buckets can be inferred based on the total ordering constraint among the buckets, which is enforced at every time step. The total ordering constraint implies that at every time step, all W buckets appear before the start of the first N bucket. Thus, if a bucket with no labeled member pixel has a preceding N bucket it should be assigned to N class, else to the W class.

[1] The computational complexity of topological sort algorithm is $O(V + E)$.

The pixels are then assigned the label of their bucket.

**Illustrative example** To further clarify the approach discussed above, Figure 4.5 shows a schematic for estimating depth order and final labels from correct but incomplete input image classification using the topological sort method. First, graph $G$ is constructed from the input classification by adding edges between pairs of nodes if at any time step one of them is labeled as W and other as N. For example, the edge $e_{12}$ is added due to time step $t1$. Next, to obtain the bucket order using the topological sort algorithm, we search for nodes with no incoming edges in $G$ (in this example nodes 1 and 3) and put them in the first bucket. Next, all edges from these nodes are removed from graph $G$ and the next set of nodes with no incoming edges are put in the next bucket. This process is repeated till $G$ is empty. This gives us the bucket order that corresponds to the depth contours in our application. Finally, to obtain the output classification, for each time step the instances of every bucket are inspected to identify the label for the bucket. For example, at time step $t1$, the top bucket (consisting of pixel 1 and 3) is assigned to W since pixel 1 $\in$ top bucket is classified as W at $t1$ in the input classification. Similarly, middle bucket is assigned to N as pixel 2 is assigned to N at $t1$. However, the instance of bottom bucket is not labeled in the input labels at time $t1$. But the total order among the buckets constrains that if middle bucket is N for a given time step, then all subsequent buckets (which are expected to have lower depth) must be assigned N for that time step. Hence, the bottom bucket (which contains pixel 4) is assigned to N class at time $t1$.

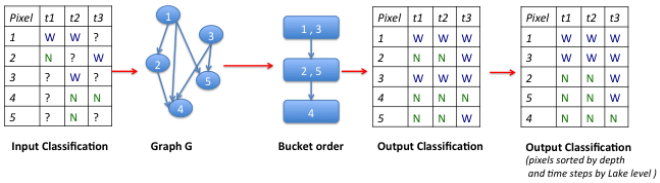



Figure 4.5: A schematic for estimating depth contours and final labels from "correct but incomplete" input image classification. The input classification product is first converted to graph $G$ and then a bucket order. The inferred bucket order together with initial input labels is then used to assign classification labels to all missing instances.

### 4.3.2 Noisy and incomplete multi-temporal image classification

*Noisy and incomplete* image classification refers to an input classification in which the class label $p_i^t$ corresponding to pixel $i$ at time step $t$ may be incorrect. Moreover, for several pairs of ($i$,$t$) the label is missing in the input classification. The goal of SELPh is to re-assign labels to all instances such that the missing labels are correctly imputed and the incorrect labels are re-assigned to their correct class in the final output classification.

**Estimating depth order** We model the pairwise information on relative depth of pixels as a directed graph $G = (V, E)$. The set of vertices ($V$) corresponds to the pixels of the image. The set of edges ($E$) capture the relationship between a pair of pixels at each time step; $e_{ij}^t$ is the edge from node $i$ to node $j$ at time $t$. We compute the graph $G$ by adding the edges $e_{ij}^t$ between pairs of nodes $i$ and $j$.

In case of *noisy* input classification there may exist pairs of locations ($i$,$j$) and time steps $(t_1, t_2)$ for which at time $t_1$ $\{p_i^{t_1} = \text{W}\ \&\ p_j^{t_1} = \text{N}\}$ that adds edge $e_{ij}^{t_1}$ to $G$, and at time step $t_2$ $\{p_i^{t_2} = \text{N}\ \&\ p_j^{t_2} = \text{W}\}$ that adds edge $e_{ji}^{t_2}$. This creates cycles in graph $G$, and due to the presence of cycles in graph $G$, there may not exist any ordering $B$, such that all the edges in $G$ are forward edges in the bucket order [63], i.e. for all edges $e_{ij}^t \in E$ the bucket of $i$ in $B$ preceeds the bucket of $j$ in $B$.

*Mathematical formulation*

We formulate the problem of estimating depth order from graph $G$ as a maximal $K$-ordered graph partitioning problem: assign the vertices to one of the $K$ ordered buckets (partitions) so as to maximize the number of edges in $G$ that agree with the ordering (*forward edges*) minus the number of edges that disagree with the ordering (*backward edges*).

Note that in reality multiple pixels may have the same depth, i.e. belong to the same depth contour. Due to such grouping structure present in data, the ordering relationship among pixels is more appropriately represented by a bucket order rather than a total order. A bucket order is a special case of partial orders in which each bucket consists of multiple entities (eg. in our case pixels) with no order among themselves and there exists a total order among the buckets.

Consider a given bucket order $B$ with $K$ buckets. The forward set $F$ of $B$ is defined

as the set of directed pairs $(i, j)$ such that $i \in B_m$ and $j \in B_n$ and $m < n$. Similarly, the backward set $R$ is defined as the set of directed pairs $(i, j)$ such that $i \in B_m$ and $j \in B_n$ and $m > n$. Then our mathematical objective for searching the maximal $K$-ordered partitioning (bucket order) can be written as

$$\max_{B} \sum_{t=1}^{T} \Big[ \sum_{(i,j)\in F} e_{ij}^{t} - \sum_{(i,j)\in R} e_{ij}^{t} \Big]$$
$$\text{s.t. } \#\text{ buckets in } B = K$$

*Algorithm*

Consider a special case of the above objective where the value of $K = 2$, i.e the goal is to split the graph into exactly two partitions. Agrawal et al. in [64] have shown that the optimal solution for this special case can be computed in $O(V + E)$ time. Specifically, they have shown that assigning the nodes into two sets- nodes of positive net degree (outdegree - indegree) in one partition and nodes of negative net degree in the other partition, gives the optimal split that maximizes the objective.

However, to the best of our knowledge, there is no existing polynomial time algorithm to find the optimal bucket order for the case $K > 2$. In fact, Agrawal et al. in [64] have also shown that the case $K = |V|$, i.e. obtaining a total ordering among nodes, is an NP hard problem.

Therefore, to solve the objective for $K > 2$, we use a heuristic approach that starts with all nodes placed in a single bucket and then iteratively increases the number of buckets by spliting one of the buckets in the current bucket order till the bucket order has the desired number of buckets (i.e. $K$). More specifically, at each iteration, the optimal split and *splitgain* (the value of the objective corresponding to optimal split) is computed for every bucket in the current bucket order. Then, the bucket that has the maximum *splitgain* is split into two, thereby increasing the size of the bucket order by one bucket at every iteration. This iterative procedure is continued till a bucket order with the desired number of buckets is reached. Note that this algoithm makes greedy (locally optimal) splits at each iteration to reach the desired number of buckets. However, the greedy strategy is only a heuristic and does not guarantee global optimality of the bucket order obtained. The detailed pseudo-code for this step is provided in the Algorithm below.

**Require:** current multi-temporal classification $X$ and *numbuckets*

$buckets \leftarrow 1$

$B_{\{1\}} \leftarrow V$ // Initially all pixels are put in single bucket

// loop till size($B$) is *numbuckets*, increasing 1 bucket at each iteration

**while** $buckets < numbuckets$ **do**

// select the bucket with maximum split gain

**for** $k = 1$ to $buckets$ **do**

// construct graph $G$ for members of $B_k$

$E \leftarrow \emptyset$

**for** $t = 1$ to $T$ **do**

**for** all $(i, j)$ location pairs of members of bucket $B_k$ **do**

**if** $\{x_i^t = \text{W}\ \&\ x_j^t = \text{N}\ \}$ **then**

$E = E \cup e_{ij}^t$

**end if**

**end for**

**end for**

// compute split gain

**for** each edge $e_{ij}^t$ **do**

$delta(i) \leftarrow delta(i) + 1$

$delta(j) \leftarrow delta(j) - 1$

**end for**

$gain(k) = sum(delta(delta > 0)$

**end for**

$splitbucket \leftarrow \text{argmax}(gain)$

// splitting $B_{splitbucket}$

$B_{j+1} \leftarrow B_j; \forall j > splitbucket$

$B_{splitbucket+1} \leftarrow$ members of $B_{splitbucket}$ with +ve *delta*

$B_{splitbucket} \leftarrow$ members of $B_{splitbucket}$ with -ve *delta*

$buckets \leftarrow buckets + 1$

**end while**

**Estimating labels** All the members of a bucket, at any given time step, will be either W or N in output classification. However, since the input classification may be incorrect, for any given time step, the *initially labeled* pixels of a bucket may have disagreeing labels.

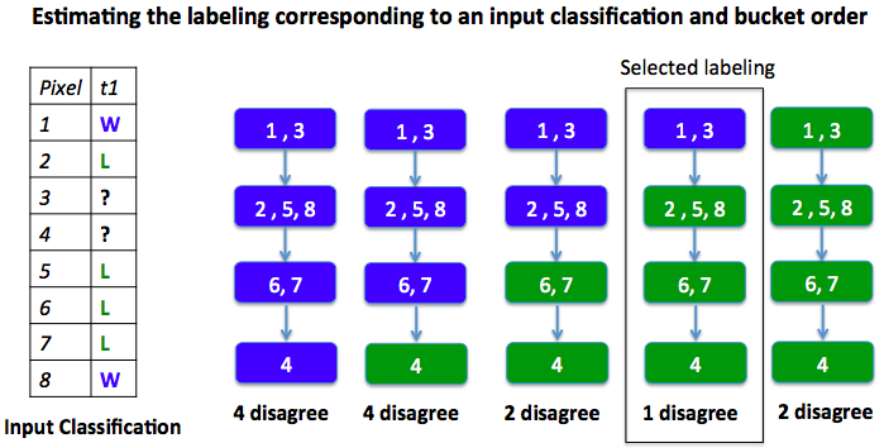



Figure 4.6: An illustrative example showing how SELPh estimates labels corresponding to a given "incorrect and incomplete" input classification and bucket order. For a given bucket order with $k$ buckets, there are $k+1$ options for labeling buckets with W (*in blue*) or N (*in green*) class, which also enforce the total order constraint. For each of these options, the number of disagreements with the input classification is computed, and the bucket labeling with minimum number of disagreements is selected.

For a given bucket order with $k$ buckets, out of the $2^k$ options for labeling buckets with W or N class, there are only $k+1$ options that enforce the total order constraint among buckets imposed by gravity. To select the bucket labeling from these $k+1$ options, at each time step, we count the number of disagreeing labels in input classification at that time step corresponding to each of these $k+1$ bucket labelings. The bucket labeling corresponding to the minimum number of disagreeing input labels is chosen. Figure 4.6 shows an illustrative example of how this step labels the buckets corresponding to a given input classification and bucket order.

Once the bucket labeling is obtained, every pixel is assigned the label of its bucket. To account for the uncertainty in the label of the boundary buckets, the pixels belonging

to the boundary buckets are re-assigned their labels in the input classification. The pseudo-code for this step is provided below.

**Require:** current multi-temporal classification $X$ and current bucket order $B$

$T \leftarrow$ number of time steps in $X$

$k \leftarrow$ number of buckets in $B$

**for** $t = 1$ to $T$ **do**

// select bucket (depth contour) at the boundary of W and N by computing disagreements for each possible boundary bucket

**for** $i = 1$ to $k$ **do**

$inwater \leftarrow \cup B_j, \forall j < i$

$inland \leftarrow \cup B_j, \forall j > i$

$disagree(i) \leftarrow \#\{X(inland, t) = \text{W }\} + \#\{X(inwater, t) = \text{N }\}$

**end for**

$boundary \leftarrow \text{argmin}(disagree)$

// update labels of $X$ at time $t$ using selected $boundary$

$inwater \leftarrow \cup B_j; \forall j < boundary$

$inland \leftarrow \cup B_j; \forall j > boundary$

$X(inwater, t) \leftarrow \text{W}$

$X(inland, t) \leftarrow \text{N}$

**end for**

**Simultaneous Estimation of Labels and Physical properties (depth ordering)** Now that we have a method to estimate bucket order $B$ from noisy input classification $P$, one option is to first get the best estimate of $B$ and then estimate the final output classification from $B$ and $P$ using the method described. Our results show that using this approach leads to significant increase in classification accuracy compared to the input $P$. In particular, the constraint on class labels imposed by the estimated bucket order helps in correctly imputing the missing labels and correcting some of the misclassifications in the input $P$.

The correctness of the estimated bucket order $B$ depends on the level of noise in $P$. Thus, for an input $P$ with a high level of noise in input labels, the bucket order obtained is likely to suffer from incorrect ordering among pixels. This incorrect ordering in $B$

impacts the accuracy of the final output- i.e. it impedes the correction of some misclassifications and even worse may lead to incorrect flipping of labels of some instances that were correctly labeled in $P$.

In SELPh we address this issue by doing a simultaneous estimation of labels (output classification) and physics (depth order). In particular, we use an iterative scheme in which instead of estimating the final bucket order at the very first iteration, the granularity of the bucket order (i.e. number of buckets in $B$) is gradually increased at every iteration. Moreover, at every iteration, an updated version of classification is obtained by improving the labels using the physics-based constraints imposed by the current bucket order. Thus, at every iteration, both the uncertainty in bucket order (inverse of the number of buckets) and the imperfection in the classification (number of missing labels and misclassified instances) is reduced. In fact, the reduction of uncertainty in $B$ (i.e. increase in number of buckets in $B$) helps in reducing the imperfections in classification by adding more physics-based constraints. Similarly, using the classification at current iteration (which has less imperfections than $P$) decreases the chance of introducing incorrect ordering among pixels as the number of buckets in $B$ is increased. Our results confirm that the iterative approach, which leverages the feedback between the estimation of labels and physics, has a significantly higher classification accuracy compared to the sequential approach. The pseudo-code for this step is provided below.

**Require:** initial multi-temporal classification $P$
  // update bucket order and class labels,
  // increasing the number of buckets in each iteration
  $X \leftarrow P$ // $X$ is the updated labeling at current iteration
  $numbuckets \leftarrow 1$
  $B_{\{1\}} \leftarrow V$ // Initially all pixels are put in single bucket
  $T \leftarrow$ number of time steps in $P$
  **while** $numbuckets < T$ **do**
    $numbuckets \leftarrow numbuckets + 1$
    $B \leftarrow \text{updateBucketOrder}(X, numbuckets)$
    $X \leftarrow \text{updateLabels}(X, B)$
  **end while**

***Note on the time complexity of SELPh:*** The SELPh Algorithm calls update label and update bucket algorithms $T$ times ($T$ is the number of time steps). Algorithm update bucket uses $O(N)$ computations for each time step ($N$ is the number of locations). Thus its time complexity is $O(NT)$. Algorithm update label uses $O(N^2)$ computations for determining edges of graph for each time step. Thus its time complexity is $O(N^2T)$. Therefore, the total time complexity for SELPh is $O(N^2T^2)$. Note that since SELPh is applied to each lake independently, the processing is highly parallel. .

## 4.4 Evaluation

In this section we analyze the performance of SELPh approach and compare it with other baseline algorithms for classification enhancement. Due to the absence of ground truth of lake surface dynamics, it is infeasible to provide quantitative evaluation on real lakes. Therefore, we provide quantitative evaluation on synthetically generated lake dynamics data along with two case studies on real lakes.

### 4.4.1 Synthetic Data Experiments

#### Generation

Here, we describe synthetic data generation process-
1) Extent and Dynamics: First, the extent for different timesteps are created such that the dynamics in the lake is physically consistent i.e. the synthetic water body grows and shrinks according to the predefined inherent ordering of locations. This set of extent maps are the ideal maps that we intend to recover after label correction. Hence, they will be used as ground truth to compare the performance of various algorithms.
2) Noise Structure: Now, noise is introduced in the ground truth extents to create the dataset that will be provided as input to different algorithms for correction. Noise can have different characteristics and hence will impact algorithms differently. Here, we have analyzed 3 types of noise structures -
**Random Noise (RN)**: (location, timestep) pairs i.e. pixels are randomly selected and noise is added in those pixels.
**Spatial Noise (SN)**: Pixels are randomly selected as seed pixels around which spatially auto-correlated noise is added. The spatially auto-correlated noise is added only

into the timestep to which that pixel belong.

**Spatio-temporal Noise (STN)**: Pixels are randomly selected as seed pixels around which spatially and temporally auto-correlated noise is added. First, strength of temporal auto-correlation is randomly selected. This determines how many timesteps around the timestep of the seed pixel would be affected by noise. Then for each of those timestep, spatially auto-correlated noise is added around seed location using the strategy described before.

### Evaluation Measure

The goal of classification enhancement is to correct the noisy labels (i.e. misclassified instances) without incorrectly flipping the labels of the correctly classified instances in the input. We compare the performance of different classification enhancement techniques using the following performance measure:

$n_A$ = number of misclassified instances after algorithm $A$

$n_0$ = number of misclassified instances in input

Performance(algorithm $A$) = 1 - $\frac{n_A}{n_0}$

### Results

**Role of simultaneous estimation** One of the key intuition behind the SELPh approach is that direct inference of the physical properties from imperfect classification may lead to incorrect estimates, and that a framework that simultaneously optimizes the two tasks of (i) inferring the physical properties and (ii) classification enhancement is likely to provide better estimation. To evaluate this hypothesis, we compare the performance of the direct inference based approach (SEQ) with SELPh for all three label noise types in Table 4.1 (40% noise was added to the input labels). The results clearly indicate that simultaneous estimation considerably improves the classification enhancement performance.

**Comparison with traditional spatial and temporal filtering schemes** Spatial and temporal smoothing approaches are widely used in remote sensing image analysis for classification enhancement. We applied the simple spatial majority (SS) and temporal majority (TS) filters for classification enhancement. The width of the filters was varied

| | *RN* | *SN* | *STN* |
|---|---|---|---|
| SEQ | 73 | 70 | 68 |
| SELPh | 90 | 85 | 80 |

Table 4.1: Comparison of SELPh with single step SEQ approach for 40% noise added to input labels.

and we report results for the width parameter that gave the best results. Our results in Table 4.2, 4.3 and 4.4 demonstrate that SELPh shows a better performance compared to majority filters in presence of high levels of noise in classification, especially in context of spatially and temporally auto-correlated noise. In fact, we observe in Tables 4.3 and 4.4 that spatial filtering scheme $SS$, which is one of the most commonly used approach in de-noising of remote sensing images, shows an extremely poor and erratic performance when encountered with spatial and spatio-temporally auto-correlated noise. This is because it is unable to correct incorrect classifications as they occur as big spatial patches and may also incorrectly smooth sharp boundaries of lakes. Temporal filtering $TS$ shows relatively better performance, especially when noise is only spatial in nature and temporally random. Finally, SELPh approach that does not assume either temporal or spatial randomness in the noise process shows the best performance on maps plagued with spatial and spatio-temporal noise.

| | *SS* | *TS* | *SELPh* |
|---|---|---|---|
| 10% | 77 | 64 | 93 |
| 20% | 85 | 50 | 90 |
| 40% | 57 | 15 | 89 |

Table 4.2: Performance of different classification enhancement strategies for random noise process.

| | *SS* | *TS* | *SELPh* |
|---|---|---|---|
| 10% | 7 | 70 | 90 |
| 20% | 8 | 55 | 88 |
| 40% | 3 | 25 | 84 |

Table 4.3: Performance of different classification enhancement strategies for spatial noise process.

| | *SS* | *TS* | *SELPh* |
|---|---|---|---|
| 10% | 7 | 40 | 91 |
| 20% | 9 | 30 | 87 |
| 40% | 6 | 12 | 80 |

Table 4.4: Performance of different classification enhancement strategies for spatio-temporal noise process.

**SELPh performance improves with increase in number of images** The efficacy of SELPh lies in the ability to reconstruct the elevation ordering despite presence of noise in the initial classification product. It is obvious that the accuracy of the estimated elevation order depends on the level of noise in the input classification. However, we observe that for the same level of noise in the input classification, the performance of SELPh significantly improves as the number of time steps (images) is increased (see Table 4.5). This is due to the fact that during the estimation of the depth order, our algorithm can leverage the additional information present in a larger set of images because *elevation remains fixed across all time steps.* Traditional spatial and temporal methods that only consider local (spatially and temporally) context for classification enhancement fail to exploit information in additional images, and their performance is relatively independent of the number of images available.

| | *50* | *100* | *200* | *500* |
|---|---|---|---|---|
| SELPh | 37 | 60 | 78 | 90 |
| SS | 9 | 9 | 8 | 9 |
| TS | 20 | 21 | 20 | 20 |

Table 4.5: Impact as the number of time steps is increased from 50 to 500 on different classification enhancement strategies (for 40% spatio-temporal noise process).

**Sensitivity to number of buckets** The SELPh algorithm requires a user-specified parameter- the number of buckets (i.e. the number of depth contours) to be created for a lake. In principle, each contour corresponds to a unique water level for the lake. Since we are working with a finite number of satellite images and each image can only provide at most one unique water level, the number of bucket is upper bounded by the number

of time steps. Therefore, in our method we set the number of buckets as the number of time steps for which the classified images are available. However, in practice, due to temporal and seasonal auto-correlations, the number of unique water levels are typically lower than the total number of time steps. This is reflected in Table 4.6 that shows the variation in the performance of SELPh with the choice of number of buckets. We observe that SELPh performance first increases rapidly with increase in the number of buckets used (till a point where the number of buckets reach the underlying number of unique water levels for the lake) and then it remains constant with any further increase in the number of buckets.

| | *50* | *100* | *150* | *200* | *250* | *300* | *500* |
|---|---|---|---|---|---|---|---|
| SELPh | 44 | 68 | 81 | 87 | 89 | 90 | 90 |

Table 4.6: Impact of increasing the number of buckets from 50 to 500 (for 40% spatio-temporal noise process using 500 time steps).

### 4.4.2 Case Study: Lake Abbe, Ethiopia

Figure 4.7 shows the initial classification maps (middle column) of Lake Abbe for three different time steps. Each pixel is classified into either the water (*in yellow*) or land (*in blue*) land cover classes. Figure also shows the SELPh output (right column) for the three images. The key differences between the initial classification and SELPh ouput are highlighted using a circle marker. To verify whether the re-assignment of label done by SELPh is correct or not, Figure 4.7 shows the multispectral false color composite (left column) for the three dates. The imagery in the spectral composites shows that the initial classification suffered from some misclassifications, and that the label re-assignments done by SELPh are indeed correct (in agreement with the water body outline visible in spectral composites).

It is also important to observe that the errors in initial classification are spatially auto-correlated. For example, the errors inside the red circle in the middle row and the black circle in the bottom row clearly show entire patches being misclassified. Furthermore, these errors were also temporally auto-correlated, i.e. persisted for multiple consecutive time steps.

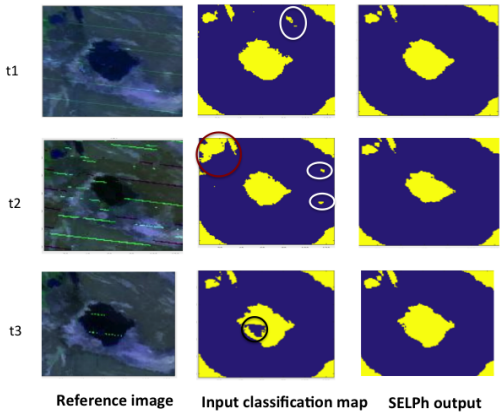



Figure 4.7: Illustration of SELPh on Lake Abbe, Ethiopia for three dates. The figure shows that spatially auto-correlated noise is corrected using SELPh. The changes made by SELPh can be verified using the spectral false color images provided for reference.

Finally, note that the surface of this lake water body is quite dynamic and all three time steps differ in their surface area. The lake was medium-sized in the early years (time step t1), shrank considerably in the middle years (time step t2), and finally grew in size during the last few years (time step t3). If SELPh is not applied on the initial classified images, due to the misclassifications in these images the exact behavior of the lake surface can be misunderstood. For example, there is a large patch that is misclassified as land in the middle of the lake at time t3. If we estimate the lake surface area for time step t3 from the classified image, we will get an incorrect estimate due to this misclassified patch.

### 4.4.3 Comparison with Profile Matching approach

Another post-classification refinement method, which can also be viewed as an instantiation of the SELPh approach, was presented in [65]. This method assumes an implicit ordering among all instances that can be used to correct the misclassified labels. Since the ordering is not explicitly given as input, the algorithm starts with an initial ordering based on a simple heuristic (eg. random ordering). Then, it iteratively improves the estimation of the ordering by matching the label profile of each pixel (i.e. the label

assignment sequence in the input classification) to the estimated profiles of different depth contours (i.e. the refined label assignment for each depth contour), and assign the pixel to the depth contour that is most similar to its label profile.

The ability of the SELPh approach to iteratively improve the estimates of the physical properties and final classification rests on certain assumptions on the nature of the imperfections present in the input classification. If these assumptions are violated, then the performance of that method can get impacted. For example, the SELPh approach assumes that the probability of error in the input classification is less than 0.5, i.e. $P(p_i^t = W|y_i^t = L) < 0.5$ and $P(p_i^t = L|y_i^t = W) < 0.5$. The post-classification refinement method presented in [65] also makes the above assumption. Moreover, the profile matching (PM) method in [65] uses a similarity function to orders pixels, which additionally assumes that there is no missing data and that the probability of error in the two classes is identical, i.e. $P(p_i^t = W|y_i^t = L) = P(p_i^t = L|y_i^t = W)$. However, in practice the classification maps typically show class conditional noise, i.e. probability of error in one of the classes is greater than the other. In our experiments we observed that these two elevation-ordering based approaches showed similar performance on data sets where $P(p_i^t = W|y_i^t = L) = P(p_i^t = L|y_i^t = W)$, but the approach presented has a considerably better performance in presence of class conditional noise compared to the profile matching method as seen in Table 4.7.

| | *RN* | *SN* | *STN* |
|---|---|---|---|
| PM | 72 | 70 | 65 |
| SELPh | 84 | 83 | 78 |

Table 4.7: Comparison of SELPh with PM approach for class conditional noise.

## 4.5 Limitations of the approach

In this section we discuss some of the limitations of our approach for correcting misclassifications of the original input classification maps.

The method requires certain degree of randomness in the noise process in order to infer depth and perform classification enhancement. It is less effective in cases where there is systematic class confusion, i.e. certain patches of land are regularly misclassified

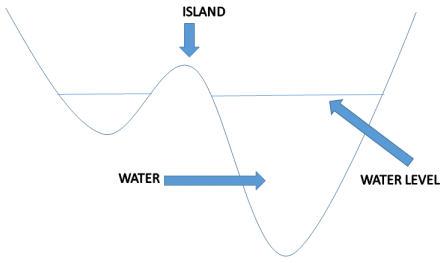



Figure 4.8: Lake with multiple concave surfaces.

due to bias in the classifier. For example, if a certain type of vegetation is misclassified as water at all time steps, then SELPh approach will fail to correct such errors.

The SELPh approach requires that all pixels of the lake upto a certain height are *water*, and all others are *land*. For example, the lake shown in Figure 4.8 consists of two concave surfaces. If the water fills to same height in the two concave bowls at all time steps, then SELPh approach is able to correct the misclassifications. However, if it happens that the water height is different for the two bowls, then SELPh performance degrades. To address this, we need to first pre-process the data to separate different lake bodies and then apply SELPh on each lake independently.

If the class label for a pixel is unobserved throughout the entire time period, then there is no information to estimate the elevation of this pixel and the presented SELPh method does not assign any label to such pixels. Since elevation field exhibits certain degree of spatial smoothness, future extensions can potentially leverage elevation estimates of nearby pixels in order to estimate elevation for such pixels. However, in practice such a situation occurs rarely.

Finally, our algorithm assumes that the elevation of a location remains fixed across all time steps. In some cases, eg. erosion or volcanic and earthquake activity, it is possible that the elevation of pixels change over time thereby changing the lake geometry. In such cases one should apply SELPh on shorter temporal windows in which the probability of elevation changes is much smaller.

## 4.6 Concluding remarks

SELPh is a new physics-guided classification enhancement algorithm to improve multi-temporal raster maps of lake water bodies by leveraging the constraints enforced by the law of gravity. SELPh is able to correct errors much more effectively than other techniques such as temporal and spatial filtering that do not take into account physical properties. In addition to reducing misclassifications, SELPh also correctly imputes missing labels. SELPh is one of the concepts that is being used to produce global-scale product of lake dynamics for 2001-2015 using MODIS data. An early version of this product is publicly available at the following url: `z.umn.edu/monitoringwater`.

The SELPh approach can be extended along several directions. The current formulation assumes that the quality of each time step is same. This is not true in practice as some time steps are more noisy and/or have more missing data than others due to atmospheric conditions and angle of the sun at the time of satellite overpass. Modeling the quality of each image of the multi-temporal stack can reduce the impact of poor quality images. Similarly, the some locations tend to be more noisy than others and impact the depth ordering. Exploring hybrid approaches that leverage spatial and temporal context in addition to physical properties to address these issues is of interest. SELPh assumes that the two probabilities of error: $Pr(y = W|x = L)$ and $Pr(y = L|x = W)$ are equal. The proposed model can be extended to incorporate notion of class conditional noise. Moreover, incorporating spatial and temporal context, in addition to physics-guided constraint, in the classification enhancement framework is expected to further improve the final classification enhancement output. Another direction of research is to integrate depth-guided constraints during the training of the original classification model and while predicting.

# Chapter 5

# Conclusion and Future Directions

In this thesis we explored the problem of identifying rare events in the presence of noisy and missing data. The first challenge addressed in this thesis is to learn classifiers when the samples available for training have considerable label noise. In remote sensing domain absence of high quality training samples is common due to heterogeneity in spatio-temporal data, and therefore it is imperative to develop algorithms to train predictive models with such imperfectly labeled samples in order to enable global scale studies. The second challenge addressed in this thesis is to identify land cover change events from multi-temporal spatial raster images when the available remote sensing data sets are plagued with noise and missing data because of obfuscation due to clouds and other atmospheric disturbances. Next, we summarize the contributions made in this thesis and some future research directions.

## 5.1 Learning predictive models for identifying rare events using imperfect training labels

In this thesis, we presented an approach to learn predictive models using imperfect labels for rare classes, and show that if the imperfect labels satisfy certain assumptions, it is possible to optimize for precision and recall of the rare class. Furthermore, we showed that if the imperfect labels are available for all instances the precision and recall of the predictions can be further improved. This approach allows us to address issues created by widespread heterogeneity and noise in remote sensing data sets. In particular, it

helped in development of a database of forest fires using remotely sensed signal using only imperfect labels for training.

A major limitation of the approach presented in this thesis is the class conditional label noise (CCN) assumption. In many domains the imperfect labels available may not satisfy the CCN assumption. This may create problems as the model trained using such imperfect labels are likely to learn incorrect class boundaries due to the noise in labels. Further research is needed to develop methods that are robust to presence of label noise that is dependent on the attributes. Another research direction is to use multiple annotators to overcome the CCN assumption on the imperfect labels. There is existing work on multi-annotator frameworks that build a *consensus* classification model by leveraging the collective information from multiple annotators and modeling annotator-specific subjectivity [66, 67].

Bringing elements of other machine learning paradigms such as active learning, multi-view learning and multi-task in the framework will also be of interest. As an example, partitioning of data to create homogeneous groups that are likely to satisfy CCN assumption may result in scarcity of positive samples in some groups. Multi-task learning can be used for sharing knowledge across related groups to address this issue.

Finally, there are opportunities to exploit any available gold standard labeled samples to improve the model trained with imperfect labels. For example, a small set of gold standard labeled samples can be used to initialize model training, which can then be further refined using the large number of imperfectly labeled samples. Another possibility is to use gold standard labeled samples for assessing the performance of the models trained using imperfect labels.

## 5.2 Classification enhancement using physics-guided properties in spatio-temporal data

In this thesis, we discussed classification enhancement algorithms to address the issue of errors in classification maps due to spatial heterogeneity, seasonal changes and variations in atmospheric conditions such as clouds and aerosols. We focus on applications where the classification output is constrained by some physical properties of the phenomena

under consideration and present approaches that can leverage such constraints to address the limitations of traditional classification enhancement techniques. In particular, we presented a temporal modeling classification enhancement algorithm that makes use of the fact that urban growth is rare and persistent to improve multi-temporal urban growth maps. Similarly, we presented an iterative algorithm that makes of the constraints on water level in lakes imposed by elevation profile to improve the land/water classification maps.

The approaches presented can be extended along several directions. The current formulations assume that the quality of each time step is same. This is not true in practice as some time steps are more noisy and/or have more missing data than others due to atmospheric conditions and angle of the sun at the time of satellite overpass. Modeling the quality of each image of the multi-temporal stack can reduce the impact of poor quality images. Similarly, the some locations tend to be more noisy than others. Exploring hybrid approaches that leverage spatial and temporal context in addition to physical properties to address these issues is of interest.

Furthermore, the algorithms can be extended to incorporate the spatial and temporal heterogeneity characteristics of the phenomena. This would require maintaining separate model parameters for data partitions corresponding to different seasons and geographical regions. Ideas on sharing model parameters can be used to address issues that may arise due to sparsity of change events in data created as a result of data partitioning.

# References


[1] Varun Mithal, Ashish Garg, Shyam Boriah, Michael Steinbach, Vipin Kumar, Christopher Potter, Steven Klooster, and Juan Carlos Castilla-Rubio. Monitoring global forest cover using data mining. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(4):36, 2011.

[2] Varun Mithal, Ashish Garg, Ivan Brugere, Shyam Boriah, Vipin Kumar, Michael Steinbach, Christopher Potter, and Steven A. Klooster. Incorporating natural variation into time series-based land cover change detection. In *Proceedings of the 2011 NASA Conference on Intelligent Data Understanding (CIDU)*, pages 45–59, 2011.

[3] Varun Mithal, Guruprasad Nayak, Ankush Khandelwal, Vipin Kumar, Nikunj C Oza, and Ramakrishna Nemani. Rapt: Rare class prediction in absence of true labels. *IEEE Transactions on Knowledge and Data Engineering*, 29(11):2484–2497, 2017.

[4] Varun Mithal, Guruprasad Nayak, Ankush Khandelwal, Vipin Kumar, Ramakrishna Nemani, and Nikunj C Oza. Mapping burned areas in tropical forests using a novel machine learning framework. *Remote Sensing*, 10(1):69, 2018.

[5] Varun Mithal, Ankush Khandelwal, Shyam Boriah, K Steinhauser, and Vipin Kumar. Change detection from temporal sequences of class labels: Application to land cover change mapping. In *SIAM International Conference on Data mining, SDM. SIAM*. SIAM, 2013.

[6] Aditya Menon et al. Learning from corrupted binary labels via class-probability estimation. In *In Proc. of the Int. Conf. in Machine Learning (ICML)*, pages 125–134, 2015.

[7] Nagarajan Natarajan et al. Learning with noisy labels. In *NIPS*, pages 1196–1204, 2013.

[8] Nitesh V Chawla et al. Smoteboost: Improving prediction of the minority class in boosting. In *Knowledge Discovery in Databases*, pages 107–119. Springer, 2003.

[9] David MJS Bowman, Jennifer K Balch, Paulo Artaxo, William J Bond, Jean M Carlson, Mark A Cochrane, Carla M DAntonio, Ruth S DeFries, John C Doyle, Sandy P Harrison, et al. Fire in the earth system. *Science*, 324(5926):481–484, 2009.

[10] L. Giglio. Modis collection 5 active fire product user's guide version 2.4. *Science Systems and Applications, Inc*, 2010.

[11] L. Giglio, T. Loboda, D.P. Roy, B. Quayle, and C.O. Justice. An active-fire based burned area mapping algorithm for the MODIS sensor. *Remote Sensing of Environment*, 113(2):408–420, 2009.

[12] Anuj Karpatne, Ankush Khandelwal, Shyam Boriah, and Vipin Kumar. Predictive learning in the presence of heterogenity and limited training data. In *Statistical Analysis and Data Mining*. SIAM, 2014.

[13] Land Processes Distributed Active Archive Center. `http://lpdaac.usgs.gov`.

[14] Karen C. Seto et al. Global forecasts of urban expansion to 2030 and direct impacts on biodiversity and carbon pools. *PNAS*, 2012.

[15] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, pages 92–100. ACM, 1998.

[16] Zachary Lipton et al. Optimal thresholding of classifiers to maximize f1 measure. In *MLKDD*, pages 225–239. Springer, 2014.

[17] Ye Nan, Kian Ming Chai, Wee Sun Lee, and Hai Leong Chieu. Optimizing f-measure: A tale of two approaches. *arXiv:1206.4625*, 2012.

[18] Harikrishna Narasimhan, Rohit Vaish, and Shivani Agarwal. On the statistical consistency of plug-in classifiers for non-decomposable performance measures. In *NIPS*, pages 1493–1501, 2014.

[19] Pang-Ning Tan et al. *Introduction to data mining*, volume 1. Pearson Addison Wesley Boston, 2006.

[20] Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *SIGKDD*, pages 213–220. ACM, 2008.

[21] Xiaoli Li and Bing Liu. Learning to classify texts using positive and unlabeled data. In *IJCAI*, volume 3, pages 587–592, 2003.

[22] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S Yu. Building text classifiers using positive and unlabeled examples. In *ICDM*, pages 179–186. IEEE, 2003.

[23] E Ted. Multi-stage classification. In *ICDM*, pages 8–pp. IEEE, 2005.

[24] Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied statistics*, pages 20–28, 1979.

[25] Padhraic Smyth et al. Inferring ground truth from subjective labelling of venus images. 1995.

[26] Jacob Whitehill et al. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *NIPS*, pages 2035–2043, 2009.

[27] Prithviraj Sen et al. Collective classification in network data. *AI magazine*, 29(3):93, 2008.

[28] Xi C Chen, Anuj Karpatne, Yashu Chamber, Varun Mithal, Michael Lau, Karsten Steinhaeuser, Shyam Boriah, Michael Steinbach, Vipin Kumar, Christopher S Potter, et al. A new data mining framework for forest fire mapping. In *Conference on Intelligent Data Understanding (CIDU), 2012*, pages 104–111. IEEE, 2012.

[29] D. Potere et al. Mapping urban areas on a global scale: which of the eight maps now available is more accurate? *IJRS*, 2009.

[30] Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, volume 3, pages 448–455, 2003.

[31] United Nations Environment Programme (UNEP). *Keeping Track of Our Changing Environment: From Rio to Rio+20 (1992-2012).* 2011.

[32] U.N. Department of Economic and Social Affairs. World urbanization prospects. `http://esa.un.org/unpd/wup/`.

[33] Marina Alberti. The Effects of Urban Patterns on Ecosystem Function. *IRSR*, 2005.

[34] E. Bartholomé and AS Belward. Glc2000: a new approach to global land cover mapping from earth observation data. *IJRS*, 2005.

[35] C.D. Elvidge et al. Global distribution and density of constructed impervious surfaces. *Sensors*, 2007.

[36] CIESIN. Global rural-urban mapping project (GRUMP). *CIESIN, Columbia University*, 2004.

[37] A. Schneider et al. A new map of global urban extent from MODIS satellite data. *Environmental Research Letters*, 2009.

[38] Pall Oskar Gislason, Jon Atli Benediktsson, and Johannes R. Sveinsson. Random forests for land cover classification. *Pattern Recognition Letters*, 27(4):294–300, 2006.

[39] C. Huang, L. S. Davis, and J. R. G. Townshend. An assessment of support vector machines for land cover classification. *IJRS*, 23(4):725–749, 2002.

[40] H. Taubenböck et al. Monitoring urbanization in mega cities from space. *RSE*, 2012.

[41] J. Suutala et al. Discriminative temporal smoothing for activity recognition from wearable sensors. *Ubiquitous Computing Systems*, 2007.

[42] E. Jaser et al. Temporal post-processing of decision tree outputs for sports video categorisation. *SSSPR*, 2004.

[43] F.P. Nava, A.P. Nava, J.M.G. Lamolda, and M.F. Redondo. Change detection in remote sensing images with graph cuts. In *Proceedings of SPIE,*, pages 59820Q–1. Society of Photo-Optical Instrumentation Engineers, 2005.

[44] A.B. Salberg and OD Trier. Temporal analysis of forest cover using hidden markov models. In *Proceeding of IGARSS*, pages 2322–2325. IEEE, 2011.

[45] N. Viovy and G. Saint. Hidden markov models applied to vegetation dynamics analysis using satellite remote sensing. *Geoscience and Remote Sensing, IEEE Transactions on*, 32(4):906–917, 1994.

[46] N. Keshava and J.F. Mustard. Spectral unmixing. *Signal Processing Magazine, IEEE*, 19(1):44–57, 2002.

[47] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Readings in speech recognition*, 1989.

[48] U.S. Geological Survey. Landsat missions. `http://landsat.usgs.gov/`.

[49] Joana Barros. *Urban Growth in Latin American Cities*. PhD thesis, University College London, 2004.

[50] Charles J Vörösmarty, Pamela Green, Joseph Salisbury, and Richard B Lammers. Global water resources: vulnerability from climate change and population growth. *science*, 289(5477):284–288, 2000.

[51] Yoshihiro Shibuo, Jerker Jarsjö, and Georgia Destouni. Hydrological responses to climate change and irrigation in the aral sea drainage basin. *Geophysical Research Letters*, 34(21), 2007.

[52] JS Famiglietti, A Cazenave, A Eicker, JT Reager, M Rodell, and I Velicogna. Satellites provide the big picture. *Science*, 349(6249), 2015.

[53] Anuj Karpatne, Ankush Khandelwal, Xi Chen, , Varun Mithal, James Faghmous, and Vipin Kumar. Global monitoring of inland water dynamics: State-of-the-art, challenges, and opportunities. In J. Lssig K. Morik and K. Kersting, editors, *Computational Sustainability (accepted)*. Springer, 2016.

[54] EA Sproles, SG Leibowitz, JT Reager, PJ Wigington Jr, JS Famiglietti, and SD Patil. Grace storage-runoff hystereses reveal the dynamics of regional watersheds. *Hydrology and Earth System Sciences*, 19(7):3253–3272, 2015.

[55] Xi C Chen, James H Faghmous, Ankush Khandelwal, and Vipin Kumar. Clustering dynamic spatio-temporal patterns in the presence of noise and missing data. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 2575–2581. AAAI Press, 2015.

[56] Huilin Gao, Charon Birkett, and Dennis P Lettenmaier. Global monitoring of large reservoir storage from satellite remote sensing. *Water Resources Research*, 48(9), 2012.

[57] Anuj Karpatne and Vipin Kumar. Adaptive heterogeneous ensemble learning using the context of test instances. In *2015 IEEE International Conference on Data Mining, ICDM 2015, Atlantic City, NJ, USA, November 14-17, 2015*, pages 787–792, 2015.

[58] Xin Huang, Qikai Lu, Liangpei Zhang, and Antonio Plaza. New postprocessing methods for remote sensing image classification: A systematic study. 2014.

[59] Borja Rodríguez-Cuenca, Jose A Malpica, and Maria C Alonso. A spatial contextual postclassification method for preserving linear objects in multispectral imagery. *Geoscience and Remote Sensing, IEEE Transactions on*, 51(1):174–183, 2013.

[60] Konrad Schindler. An overview and comparison of smooth labeling methods for land-cover classification. *Geoscience and Remote Sensing, IEEE Transactions on*, 50(11):4534–4545, 2012.

[61] Anuj Karpatne, Zhe Jiang, Rangaraju Vatsavai, Shashi Shekhar, and Vipin Kumar. Monitoring land cover changes using remote sensing data: A machine learning perspective. *IEEE Geoscience and Remote Sensing Magazine, 2016 (to appear)*, 2016.

[62] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, Clifford Stein, et al. *Introduction to algorithms*, volume 2. MIT press Cambridge, 2001.

[63] Aristides Gionis, Heikki Mannila, Kai Puolamäki, and Antti Ukkonen. Algorithms for discovering bucket orders from data. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 561–566. ACM, 2006.

[64] Rakesh Agrawal, Alan Halverson, Krishnaram Kenthapadi, Nina Mishra, and Panayiotis Tsaparas. Generating labels from clicks. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 172–181. ACM, 2009.

[65] Ankush Khandelwal, Varun Mithal, and Vipin Kumar. Post-classification label refinement using implicit ordering constraints among data instances. In *International Conference on Data Mining*, 2015.

[66] Yan Yan et al. Active learning from crowds. In *ICML*, pages 1161–1168, 2011.

[67] Hamed Valizadegan et al. Learning classification models from multiple experts. *Journal of Biomedical Informatics*, 46(6):1125–1135, 2013.